القسم الاول

أساسيات الاتصالات المحمولة

الكلمات المفتاحية:

محمول، لاسلكي، تقنية، تجهيزات، مزود الخدمة.

ملخص:

تُعتَبَر التقنيات المحمولة موضوع اهتمامٍ كبيرٍ للمطورين لما لها من انعكاس على التطبيقات التي يمكن أن تُستَثمَر فيها. سنحاول في هذه الجلسة التعرف على المفاهيم الأساسية للتقنيات المحمولة وتطبيقاتها في التجارة والأعمال الالكترونية.

أهداف تعليمية:

يتعرف الطالب في هذا الفصل على:

- مفهوم التقنيات المحمولة واللاسلكية
- التجهيزات المحمولة، أنواعها ومواصفاتها
- مفهوم التجارة الالكترونية باستخدام التجهيزات المحمولة ومعيقاتها.

التقنيات المحمولة واللاسلكية

تعريف:

يختلف تعريف مصطلحي المحمول واللاسلكي من شخصٍ إلى آخر ومن منظومةٍ إلى أخرى.
يُستخدَم هذان المصطلحان عادةً بطريقةٍ تبادلية بالرغم من كونهما يُعبران عن تقنيتين مختلفتين.

التقنيات المحمولة هي أية تقنية يمكن استخدامها أثناء الحركة وهي تتدرج من الحواسب المحمولة إلى الهواتف الخليوية بمعنى أنه مادمنا لا نتحدث عن ثبات في الموقع فنحن نتحدث عن تقنية محمولة.

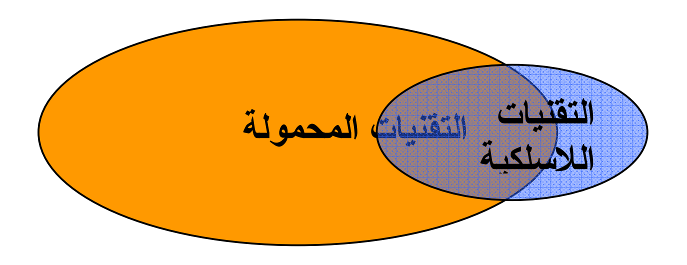
في حين يُعبِّر المصطلح (لاسلكي) عن عملية الإرسال لصوت أولبيانات باستخدام موجات راديوية. إذ تسمح التقنيات اللاسلكية للعاملين في مؤسسةٍ ما مثلاً بالوصول إلى بيانات المؤسسة دون الحاجة إلى اتصال فيزيائي ملموس مع الشبكة. بالنتيجة، تشكل التجهيزات التي تتصل ببعضها البعض بهذا الأسلوب ما يُسمى الشبكة اللاسلكية.

إذاً يمكننا القول أن التقنيات اللاسكلية تشمل كل ما يمكن استخدامه لإرسال أواستقبال البيانات عبر الشبكة اللاسلكية.

يمكن الاتصال بالشبكات اللاسكلية باستخدام تقنية محمولة أومن موقع ثابت.

كما نرى في معظم الأحيان تعتبر التقنيات اللاسلكية مجموعة جزئية من التقنيات المحمولة ولكن في بعض الأحيان يمكن لأحد التجهيزات أن يُمثل تقنيةً محمولة دون أن يكون لاسلكياً. يوضح الشكل التالي العلاقة بين التقنيات المحمولة واللاسلكية.

.



مكونات بيئة العمل اللاسلكية

تتطلب عملية تأمين بيئة عمل لاسلكية تضافر مجموعة من الخدمات والتطبيقات من جهة وتدخُّل أطراف من جهة أخرى بحيث تُشكل هذه الأطراف بمجملها سلسلة مكونات بيئة العمل.

يظهر الشكل التالي هذه المكونات:

مصنعوا التجهيزات | مكاملوا الأنظمة | بائعوا البرمجيات | مزودوا البنية | بائعوا العتاد الصلب | المشغّل اللاسلكي ومزودو الخدمة

المُشغِّل اللاسلكي ومزود الخدمة:

يتطلب أي حل لاسلكي مُشغِّل أومُزوِّد خدمة، وهوما يتمثل بشركة تؤمن البنية التحتية للاتصال عبر شبكة لاسلكية عريضة أومحلية. تكون أغلب هذه الشركات هي نفسها الشركات التي تقدم خدمات الاتصال للهواتف المحمولة وللاتصالات الصوتية.

بائعوا العتاد الصلب:

تحتاج الحلول اللاسلكية عادةً بالإضافة إلى التجهيزات المحمولة، عتاداً صلباً إضافياً. يشمل هذا العتاد تجهيزات مثل الموائمات اللاسلكية وبطاقات الشبكة ونقاط الوصول اللاسلكية.

بدأت مؤخراً محاولة تضمين التجهيزات المحمولة هذا العتاد بشكل مبيت ولكن سيلزم بعض الوقت لتصبح هذه العملية قياسية في كل التجهيزات المحمولة.

مزودوا البنية التحتية البرمجية:

يوفر مزود الخدمة والعتاد بيئة فيزيائية للعمل ولكن الحلّ لا يتكامل دون وجود بيئة برمجية للعمل. فمن الضروري مثلاً وجود متصفح انترنت، وبرنامج مخدم تطبيقات لاسلكية، وتطبيق لإدارة عمليات التخزين وغيرها، حتى نحصل على حلّ متكامل.

مكونات بيئة العمل اللاسلكية

(تتمة)

بائعوا البرمجيات المستقلون:

إضافةً إلى برمجيات البنية التحتية، تُقدم العديد من الشركات برمجيات خاصة بخدمة أوغرض معين مثل برمجيات حساب الضرائب، وبرمجيات الخدمات المصرفية، وبرمجيات الخدمات الصحية،... وغيرها.

مكاملوا الأنظمة :

وهم مجموعة الشركات التي تساعد في تقديم حزمة متكاملة من المكونات السابقة الذكر، فعلى سبيل المثال تركز العديد من هذه الشركات على تقديم الحلول اللاسلكية المحمولة المتكاملة.

مصنعوا التجهيزات:

تُعتبر التجهيزات المحمولة متطلباً رئيسياً في حلول التقنيات المحمولة. ويتوقف اختيار التجهيزات المناسبة على نوع التطبيقات المُراد التعامل معها، حيث تُعتبر أجهزة PDA وأجهزة الحواسب المحمولة أكثر ملائمةً في حلول الأعمال المحمولة من الهواتف المحمولة، لكونها تكون قادرة على تشغيل تطبيقات أكثر تعقيداً ولكونها تمتلك واجهةً ملائمةً أكثر لطبيعة هذه التطبيقات (سيتغير هذا الوضع مع ظهور الهواتف المحمولة الذكية والتي دمجت طبيعة عمل أجهزة PDA مع مميزات الهاتف العادي).

ملاحظة:

لا تكون جميع هذه المكونات مطلوبة في كافة الحلول المحمولة، إذ يمكن استخدام مجموعة جزئية فقط من مكونات هذه السلسلة لتقديم بعض الحلول.

التجهيزات المحمولة

يقدم سوق التجهيزات المحمولة طيفاً واسعاً من التجهيزات المحمولة والموجهة لتغطية الأنواع المختلفة من التطبيقات والمستخدمين. تتدرج هذه التجهيزات في الكلفة والحجم وتغطي أنواع الحواسب المحمولة وأجهزة PDA والهواتف الذكية... وغيرها.

تكتسب التجهيزات المحمولة أهميتها من كونها الجزء الوحيد من الحلول المحمولة الذي يكون على تماسٍ مباشر مع المستخدم. لذا يجب، عند تصميم أي حلٍ محمول، مراعاة خصائص هذه التجهيزات من حيث آلية الاتصال، ونظام التشغيل، وآليات الإدخال، ومستوى الأداء.

كما يجب مراعاة النقاط التالية:

1- حجم الجهاز ووزنه
2- سرعة المعالجة
3- مواصفات الشاشة من حيث الحجم والألوان وإمكانية الاستخدام الداخلي أوالخارجي
4- إمكانية تحديث نظام التشغيل
5- إمكانية وصل الطرفيات والتوسُّع
6- عمر البطارية
7- التقنيات المُدمَجة في الجهاز كوجود كاميرا أولوحة مفاتيح، أوأشعة تحت حمراء، أواتصال لاسلكي من نمط بلوتوث
8- الدعم البرمجي بما يتضمن توفر برمجيات مُطوَّرة من شركات مختلفة عن الجهة المصنعة.

التجهيزات المحمولة
(تتمة)

من بين الخصائص التي رأيناها للتجهيزات المحمولة والتي يجب مراعاتها في تصميم الحلول المحمولة، تكتسب آلية الإدخال وآلية الاتصال اللاسلكية أهمية كبيرة مما سيجعلنا نناقشها بصورة أكثر تفصيلاً.

آلية الإدخال:

تعتمد آلية الإدخال التي يجب اختيارها في التجهيزات المحمولة على مستوى التفاعل المطلوب، وعلى نوع التطبيقات التي نريد التعمل معها، عموماً، تستلزم معظم تطبيقات الأعمال المحمولة كمية كبيرة من الإدخالات لذلك يجب أخذ آلية الإدخال بعين الاعتبار في هذا النوع من الحلول.

آليات الإدخال الآكثر انتشاراً هي:

- لوحة الأرقام
- الإدخال باستخدام القلم
  - لوحة المفاتيح الافتراضية

- التعرف على المحارف
- مجموعة محارف
- التعرف المباشر على خط اليد.

- الإدخال باستخدام لوحة المفاتيح
- الإدخال الصوتي.

من بين الخصائص التي رأيناها للتجهيزات المحمولة والتي يجب مراعاتها في تصميم الحلول المحمولة، تكتسب آلية الإدخال وآلية الاتصال اللاسلكية أهمية كبيرة مما سيجعلنا نناقشها بصورة أكثر تفصيلاً.

آلية الإدخال:

تعتمد آلية الإدخال التي يجب اختيارها في التجهيزات المحمولة على مستوى التفاعل المطلوب، وعلى نوع التطبيقات التي نريد التعمل معها، عموماً، تستلزم معظم تطبيقات الأعمال المحمولة كمية كبيرة من الإدخالات لذلك يجب أخذ آلية الإدخال بعين الاعتبار في هذا النوع من الحلول.

آليات الإدخال الآكثر انتشاراً هي:

- **لوحة الأرقام:**

تحتوي الهواتف المحمولة عادةً على لوحة مكونة من 12 رقماً. تُعتبر هذه اللوحة بالغة الفعالية لإدخال الأرقام ولكن الأمور تصبح أكثر إرباكاً عند الرغبة بإدخال النصوص بسبب تخصيص كل مفتاح لتمثيل ثلاث محارف. بالطبع تم تطوير تقنيات مختلفة لجعل عملية الإدخال أكثر سهولة من هذه التقنيات تقنية T9 بحيث تقوم هذه التقنية بتوقع النص المراد إدخاله عند إدخال تتالي معين للمحارف.

- **الإدخال باستخدام القلم:**

يُعتبر أحد الإنجازات المميزة ويتمثل في تمكين استخدام شاشات اللمس التي تعتمد على القلم في الإدخال. سمح هذا الأسلوب للمستخدمين بعمليات الإدخال لاتحتاج إلى لوحة مفاتيح أوإلى لوحة أرقام فيزيائية. من أكثر التقنيات شيوعاً والمستخدمة في هذا المجال:

- لوحة المفاتيح الافتراضية: حيث يجري إظهار لوحة مفاتيح على الشاشة ويمكن للمستخدم عندها إدخال المعلومات كأنه يعمل على لوحة مفاتيح حقيقية. تكمن المشكلة في هذا الحل في الحاجة إلى مساحة شاشة كبيرة نسبياً وفي محدودية سرعة الإدخال.
- التعرف على المحارف: تعمل هذه التقنية على محاولة تفسير المحارف التي تتم كتابتها بواسطة القلم على الشاشة (على مساحة محددة من الشاشة غالباً). يوفر نظام تشغيل Windows CE هذه الميزة، وهي تُعد فعالة جداً في حال النصوص الصغيرة ولكنها تصبح أقل فعالية في حال الرغبة برفع سرعة الإدخال في النصوص الكبيرة.
- مجموعة محارف Graffiti : قدمت بعض أنظمة التشغيل الخاصة بالتجهيزات المحمولة شكلاً جديداً من أشكال التعرف على المحارف وذلك باستخدام مجموعة محارف خاصة تدعى Graffiti تسمح هذه المحارف بسهولة التعرف على

الأحرف، تتطلب هذه المحارف بعض التدريب ولكنها تحقق سرعة جيدة في الإدخال بعد أن يتم التعود عليها.

- التعرف المباشر على خط اليد: تعتمد هذه التقنية على إمكانية تدريب نظام التشغيل على التعرف المباشر على خط يد المُستخدِم. تتطلب هذه العملية، حتى تكون فعالة، برمجيات وعتاد صلب أكثر تعقيدا.

الإدخال باستخدام لوحة المفاتيح:

بالرغم من كل التطورات التي قدمت أشكال مختلفة لإدخال البيانات، ما تزال لوحة المفاتيح هي الطريقة الأكثر فعالية لعملية الإدخال. يمكن أن تكون لوحة المفاتيح مُدمَجة كما في الحواسب المحمولة أوكطرفية يمكن وصلها إلى الجهاز المحمول.

الإدخال الصوتي:

تُعتبر عملية الإدخال الصوتي عملية سهلة وذات فعالية كبيرة، إذ يمكن تنفيذ بعض الأوامر الصوتية مباشرةً كتأسيس اتصال أوالبحث عن معلومات الاتصال لشخص ما.

أما في حالة التطبيقات الأكثر تعقيداً مثل تلك الخاصة بالاستعلام الصوتي أوتنفيذ المناقلات البنكية، فعادةً يمتلك هذا التطبيق الإمكانيات الكافية للإجابة على الأسئلة الصوتية المعقدة من النمط: " كيف سيكون الطقس غداً في دمشق" أومن النمط " ما هي الطائرات القادمة من دبي إلى دمشق اليوم ". غالباً ما تستخدم هذه التطبيقات المعايير الخاصة بأنظمة الاستجابة الصوتية مثل VoiceXML.

التجهيزات المحمولة
(تتمة)

آلية الاتصال اللاسلكية:

تتوافر ثلاث أنواع رئيسية للاتصال اللاسلكي من التجهيزات اللاسلكية: اتصال باستخدام وحدتين، الاتصال باستخدام التجهيزات القابلة للفصل، الإتصال المُبيَّت ضمن الجهاز.

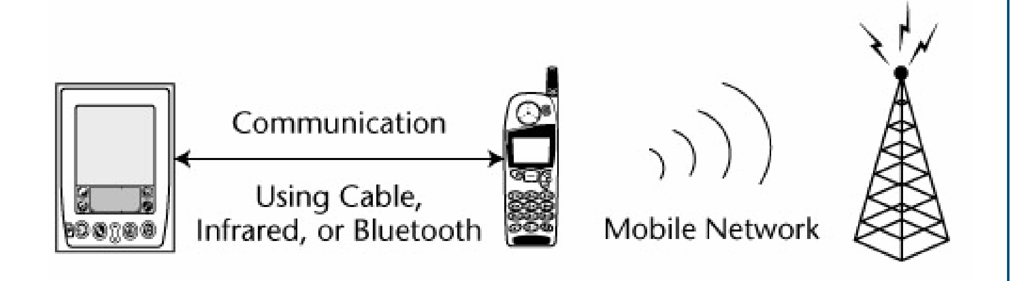
الاتصال باستخدام وحدتين:

يتطلب الاتصال من هذا النوع جهازين للعمل مع بعضهما البعض كجهاز PDA وجهاز هاتف خليوي مثلاً.

توفر أحد هذه التجهيزات الاتصال اللاسلكي للجهاز الآخر ليقوم باستعمالها.

يمكن أن تتصل هذه التجهيزات بعضها البعض بالإحدى الطرق التالية:

- الاتصال المباشر بواسطة كابل
- الاتصال باستخدام الاشعة تحت الحمراء
- الاتصال باستخدام تقنية بلوتوث.



- يساعد هذا النمط من الاتصال في اختيار الجهاز المُراد استخدامه اعتماداً على الخدمة التي صمم لأجلها ولا حاجة عندئذ للتضحية بالخدمات مقابل قدرة الاتصال لاسلكياً.
- تساعد هذه الطريقة أيضاً في استبدال الأجهزة القديمة دون الحاجة إلى استبدال النظام بكامله. كما يمكن استخدام جهاز متصل لاسلكياً لتأمين الاتصال اللاسلكي لأكثر من جهاز آخر.
- تتمثل الناحية السيئة في هذا الحل في تعقيده. إذ يجب إعداد كل من الهاتف الخليوي والجهاز المحمول لضمان عمل هذا الحل.
- كما أن عدم توفر الاتصال بتقنية بلوتوث قد يجعل من عملية الاتصال بالأشعة تحت الحمراء أوباستخدام الكابل، عملية معيقة بعض الشيء.

التجهيزات المحمولة
(تتمة)

آلية الاتصال اللاسلكية:

الاتصال باستخدام التجهيزات القابلة للفصل:

يتضمن هذا الحل استخدام الوحدات النمطية القابلة للفصل مع الجهاز المحمول لتأمين عملية الاتصال اللاسلكي. يمكن لهذه الوحدات تأمين الاتصال لأكثر من نوع من الشبكات للاسلكية بعملية إعداد بسيطة.
يتطلب استخدام هذه الوحدات توفر قابس خاص لوصل هذه الطرفيات ويكون غالباً من النمط (Compact flash) أو(PCMCI).
تتلخص الميزة الأكثر أهمية لهذا النوع من الاتصال في إمكانية استخدام وحدات مختلفة لتوفير عملية الربط مع أنواع مختلفة من الشبكات ك WLAN أو WAN أو PAN...

آلية الاتصال المبيتة ضمن الجهاز:

تعتبر عملية مكاملة أوتضمين الجهاز المحمول آلية الاتصال اللاسلكية، أحد أساليب الاتصال اللاسلكي المعتمدة.
يعتبر هذا الخيار أساسي في حال الهواتف المحمولة وقد بدأت التجهيزات المحمولة الأخرى كالحواسب المحمولة مثلاً في اعتماده مؤخراً.
تتلخص الميزة الأساسية في هذه الطريقة في تقليل التعقيد ما أمكن وتأمين تكامل مثالي بين مكون الاتصال اللاسلكي والجهاز المحمول ونظام التشغيل الخاص بالجهاز.
إلا أن فقدان المرونة يُعتبر أحد العناصر السلبية في هذا الحل، حيث يصبح مستخدم الجهاز محدوداً بنمط الشبكة اللاسلكية التي تم

مكاملة آلية الاتصال الخاصة بها ضمن جهازه

المعيارية في الحلول المحمولة

شعرت العديد من الشركات بمشكلة نقص المعايير القياسية كعائق في وجه تطوير الحلول المحمولة، وشعرت هذه الشركات بالقلق من تحول التقنيات التي تستخدمها إلى تقنيات غير معيارية حتى قبل أن تبدأ بتطوير الحلول الخاصة بها.

لذلك تم العمل على العديد من المعايير القياسية في مجال الحلول المحمولة كان من أهمها WAP الخاص بتطوير حلول التطبيقات اللاسلكية على الإنترنت حيث قام هذا المعيار بحل الكثير من المشاكل المتعلقة بالعمل بشكل موافق لمعايير الإنترنت.

كما تم تطوير لغات مختلفة مثل WML وCHTML وVoiceXML في مجال اللغات التأشيرية الخاصة بالإنترنت،

وتم استخدام تقنيتين أساسيتين هما تقنية BREW (Binary Runtime Environment for wireless) في مجال الهواتف الذكية، وتقنية J2ME من أجل طيف واسع من التجهيزات المحمولة، وذلك في مجال المعايير الخاصة بتطوير التطبيقات من جهة الزبون.

وقدمت شركة مايكروسوفت بيئة.(NETCF(.NET compact framework والتي لا تعتبر قياسية بشكل رسمي ولكنها تكتسب أهمية كبيرة كونها فتحت مجال تطوير التطبيقات للتجهيزات العاملة على نظام تشغيل windows-CE.

ولا بد في هذا السياق من ذكر كون تطبيقات C/C++ وVisual basic تعتبر خيارات جيدة عند الرغبة بتطوير التطبيقات للطرفيات الذكية.

وجرى تطوير لغة SyncML المرشحة لتكون أحد المعايير التي تستخدم لعمليات المُزامنة، إذ تسمح بعملية المزامنة لمعلومات جهات الاتصال وللمواعيد والبيانات بين الجهاز المحمول وجهاز متصل.

ولم يجر الاتفاق على معيار خاص بالـ WAN، لكن تم مؤخراً دعم بروتوكول IP بالنسبة لـ WLAN وذلك وفق المعايير 802.11a و802.11b و802.11x، وتم توفير تقنية بلوتوث معياراً للشبكات PAN.

تصنيف التجهيزات المحمولة

تتنوع التجهيزات المحمولة بشكل كبير، ويصعب تصنيفها بشكل دقيق. فيما يلي أحد التصنيفات المستخدمة لهذه التجيزات:

1- الهواتف المحمولة المجهزة بإمكانية الولوج إلى الوب.
2- أجهزة الرسائل القصيرة بالاتجاهين: تسمح هذه الأجهزة بإرسال واستقبال الرسائل النصية
3- الهواتف الذكية منخفضة السوية
4- PDA
5- الهواتف الذكية عالية السوية
6- الحواسب الشخصية اليدوية

7- الحواسب المسطحة

8- الحواسب المحمولة.

التجارة الالكترونية عن طريق الأجهزة المحمولة
m-commerce

يندرج تحت هذا النوع من التجارة الالكترونية عمليات الشراء لمنتجات أوخدمات باستخدام الأجهزة المحمولة، حيث كانت الفكرة الأساسية التي تتعلق بتطوير هذا المفهوم وأدواته، إعطاء القدرة لمستخدمي التجهيزات المحمولة على إجراء الأعمال وعمليات الشراء الالكترونية من أي مكان وفي أي وقت بنفس فعالية الأعمال الالكترونية تجريدها من قيود الاتصال السلكي.

من أهم التطبيقات لهذا المفهوم والتي يُعتقَد أنها ستقود سوق التجارة الالكترونية المحمولة خلال السنوات القادمة هي:

الشراء الرقمي: يركز هذا النوع من العملية الشرائية على المنتجات الرقمية التي يمكن تحميلها واستخدامها فوراً.

أكبر الأسواق لهذا المنتجات هي أسواق الألعاب الخاصة بالتجهيزات المحمولة، والرنات، والنغمات الخاصة بهذه التجهيزات.

التعاملات البنكية المحمولة:

هناك نوعان أساسيان من الخدمات التي يمكن تقديمها فيما يتعلق بالمعاملات البنكية.

- الأولى تللك المتعلقة بتمكين الوصول إلى الحساب البنكي للاطلاع على كشف المناقلات، إضافةً إلى إمكانية إجراء المناقلات البنكية. وتعتبر هذه الخدمة امتداداً للعمليات البنكية باستخدام الانترنت كون التجهيزات المحمولة قد تم تزويدها بإمكانية الاتصال بهذه الشبكة.
- أما الثانية فهي المتعلقة باستخدام التجهيزات المحمولة في عملية الدفع حيث يتم الدفع عن طريق الجهاز بنقد رقمي يتم تحويله إلى نقد حقيقي عن طريق الشركة التي تؤمن الاتصال كما في شركات الهاتف المحمول.

خدمات المعلومات:

يتضمن هذا النوع من الخدمات تزويد مستخدمي الأجهزة المحمولة بالمعلومات المتنوعة سواء فيما يختص بحركة سوق الأسهم المالية، أوأخبار الرياضة، أوالسياسة، أوالطقس، حيث يمكن إرسال هذه المعلومات إلى المستخدم كرسائل تنبيهية.

التجارة الالكترونية عن طريق الأجهزة المحمولة
m-commerce
(تتمة)

الخدمات المتعلقة بالموقع:

يمكن أن تُشكل إمكانية تحديد موقع المستخدم واحتياجاته أداة بالغة القوة في خدمات المبيع بحيث يتم تزويد المستخدم بالبيانات الخاصة عن المنتج الذي يحتاجه في أقرب مكان بيع فيزيائي إليه.

لكن ما يزال بعض القلق سائداً بخصوص هذا النوع من الخدمات بما يتعلق بخصوصية المستخدم ومعلومات حركته.

التسوق باستخدام التجهيزات المحمولة:

لن يكتسب أغلب أنواع التسوق الالكتروني باستخدام التجهيزات المحمولة شعبية في المستقبل القريب بسبب محدودية التجهيزات التي

تجعل التسوق بالطرق الأخرى تجربة أكثر تميزاً.
بالرغم من هذا ما تزال بعض الاستخدامات مطلوبة كما هوالحال في حجز بطاقات العروض السينمائية قبل ساعة من العرض مثلاً أوفي حالة مقارنة الأسعار المتوفرة على الانترنت لمنتج ما قبل شراءه من متجر فيزيائي.

الإعلان باستخدام التجهيزات المحمولة:

يتوفر لدى مقدم الخدمة المحمولة معلومات مختلفة حول جهاز المستخدم وموقعه والبضائع التي يقوم بشراءها وغيرها من المعلومات التسويقية.

يمكن لمزود الخدمة أن يرسل رسائل إعلانية لمنتج أومنتجات إلى المستخدمين بحسب اهتماماتهم ومستوى دخلهم وقدرتهم الشرائية.

تكمن المشكلة الأساسية في هذا النوع من الإعلانات في أن المستخدم سيقوم بتغيير مزود الخدمة في حال شعر بالإزعاج من عملية الإغراق برسائل الإعلان.

لذا يُتوقع فقط أن نرى نوعاً من الإعلانات المطلوبة من قبل المستخدم كالإعلان عن أقرب محطة وقود أوأقرب صيدلية... وغيرها.

التجارة الالكترونية عن طريق الأجهزة المحمولة

m-commerce

(تتمة)

تقع مسؤولية عدم القبول السريع للتجارة الالكترونية المحمولة على عاتق مجموعة من الأسباب الفنية والأسباب العملياتية المتعلقة بالأعمال، من أهمها:

- عدم احتواء أغلب التجهيزات والشبكات على مزايا خاصة تُسهِّل عمليات استخدام النجهيزات في عمليات الشراء
- عدم توفر واجهات استخدام مناسبة لإدخال البيانات
- عدم توفير الشبكات اللاسلكية لمزودي الخدمات (إلا مؤخراً) لسرعات كافية من أجل التعامل مع المكونات المرتبطة بإجراء عمليات الشراء.
- عدم تناسب الصور وحيدة اللون أوالصورة بالتدرجات الرمادية بشكل كبير مع عمليات الشراء. فبالرغم من أن الشركات المصنعة للتجهيزات المحمولة قد زودت تجهيزاتها بشاشات ملونة عالية الدقة، إلا أن أغلب التجهيزات المنتشرة لا تحتوي هذه الميزة.
- دم اقتناع معظم المستخدمين بأن التجارة الالكترونية (السلكية) غير آمنة لذا فما يزال هناك طريق طويل أمام التجارة الالكترونية المحمولة في موضوع الأمان حيث يدخل بالحسبان موضوع أمان إرسال المعلومات عبر الأثير كعامل خطر إضافي.
- عدم توفر التطبيقات المناسبة للتجهيزات المحمولة.
- عدم التطرق لسياسات الفوترة والتسعير بين المستهلك ومزود الخدمة من قبل صناعة التجهيزات اللاسلكية.

تشير الدلائل إلى أن مصير التجارة اللالكترونية المحمولة هوالنجاح ولكن هذا النجاح سيأخذ وقتاً أكثر مما كان متوقعاً.

القسم الثاني والثالث:

الشبكات اللاسلكية، أنواعها والتقنيات المستخدمة فيها

الكلمات المفتاحية:

مجموعة، عنصر، المجموعة الخالية، مجموعة جزئية، علاقة انتماء، علاقة احتواء، مخطط فِن، تقاطع، اجتماع، مجموعات منفصلة، المجموعة المتممة.

ملخص:

تعد الشبكات اللاسلكية أحد أكثر المواضيع سخونة في مجال التقنيات المحمولة فالجميع يتحدث عن الجيل الثالث من الشبكات اللاسلكية وعن تقنية الشبكات اللاسلكية التي ستسيطر على السوق.

سنغطي في هذا الجزء من الجلسة الأنواع الأربعة الأساسية للشبكات اللاسلكية ونتعرف على مجموعة من التقنيات المستخدمة في هذه الشبكات.

أهداف تعليمية:

يتعرف الطالب في هذا الفصل على:

- الشبكة اللاسلكية المحلية الشخصية (WPAN)
- الشبكة اللاسلكية المحلية (WLAN)
- الشبكة اللاسلكية العريضة (WWAN)
- شبكة الأقمار الصناعية (Satellite)

الشبكات اللاسلكية

تعد الشبكات اللاسلكية أحد أكثر المواضيع سخونة في مجال التقنيات المحمولة فالجميع يتحدث عن الجيل الثالث من الشبكات اللاسلكية، وعن تقنية الشبكات اللاسلكية التي ستسيطر على السوق.

سنغطي في هذا الجزء من الجلسة الأنواع الأربعة الأساسية للشبكات اللاسلكية :

1- الشبكة اللاسلكية المحلية الشخصية.(WPAN)
2- الشبكة اللاسلكية المحلية (WLAN)
3- الشبكة اللاسلكية العريضة (WWAN)
4- شبكة الأقمار الصناعية (Satellite) .

مقدمة عن الشبكات اللاسلكية

تخدم الشبكات اللاسلكية العديد من الأغراض. تُشكِّل هذه الشبكات في بعض الأحيان بديلاً عن الوصل باستخدام الكابلات بينما تُستَخدَم في أحيان أخرى لتأمين تزويد ومشاركة البيانات عن بعد.

سنقوم بتقسيم الشبكات اللاسلكية إلى نوعين رئيسين:

- **شبكات المدى القصير :** ويتناول هذا التصنيف الشبكات التي تغطي مساحة محدودة كما هي الحال بالنسبة لشبكة WLAN التي تغطي بناء، أوتجمُّع، أومعمل أوحتى منزل، كما تندرج تحت هذا التصنيف أيضاً الشبكات المحلية الشخصية WPAN.
  تعمل هذه الشبكات عادةً على ترددات ليست بحاجة إلى الترخيص ويجري حجزها للأغراض الصناعية، أوالتجارية، أوالعلمية، أوالطبية.
  تختلف الترددات المتوفرة من بلدٍ لآخر، ولكن أكثر هذه الترددات شيوعاً هو2.4 جيغاهرتز غير المحجوز تقريباً في كل أنحاء العالم. كذلك يتم استخدام الترددات 5 جيغاهرتز و40 جيغاهرتز.

- **شبكات المدى الطويل:** تُستخدَم شبكات المدى الطويل عندما لا تستطيع شبكات المدى القصير تلبية الحاجة لامتداد أكبر لتغطية الشبكة.
  غالباً ما يُستخدَم هذا النوع من الشبكات من قبل الشركات التي تقدم الاتصال اللاسلكي كخدمة. يمتد هذا النوع من الشبكات على مساحات واسعة كحال شبكة مدينة، أومحافظة، أوحتى على بلد كامل.
  أكثر أنواع الشبكات اللاسلكية طويلة المدى شيوعاً هوWWAN أوالشبكة اللاسلكية العريضة.
  أما عند الحاجة إلى اتصال لاسلكي على مستوى عالمي يمكن أن تُشَكِّل الشبكات اللاسلكية عبر الأقمار الصناعية حلاً أمثلياً.

مقدمة عن الشبكات اللاسلكية
(تتمة)

إذاً فالأنواع الأربعة الأكثر انتشاراً للشبكات اللاسلكية كما ذكرنا والتي تغطي تلك الطويلة والقصيرة المدى هي WPAN,WLAN,WWAN,Satellite وفيما يلي جدول يوضح أهم خصائص هذه الشبكات.

المعيار	سرعة النقل	الكلفة	الوظيفة	المساحة المغطاة	نوع الشبكة
IrDA, Bluetooth, 802.15	0.1-4 Mbps	منخفضة جداً	بديل عن التوصيل باستخدام الكابلات في الشبكات الشخصية	عادة حوالي 10 أمتار	(WPAN)
802.11a, b, g, HIPERLAN/2	1-54 Mbps	متوسطة	تعمل كتوسع للشبكة المحلية أوبديل عن الشبكة السلكية	عادة حوالي 100 متر	(WLAN)
GSM, TDMA, CDMA, GPRS, EDGE, WCDMA	8 Kbps-2 Mbps	مرتفعة	تعمل كتوسع للشبكة المحلية	عادة تكون التغطية على مستوى وطني باستخدام أكثر من حامل	(WWAN)
TDMA, CDMA, FDMA	2 Kbps-19.2 Kbps	مرتفعة جداً	تعمل كتوسع للشبكة المحلية	Global coverage	)Satellite(

إذا فالأنواع الأربعة الأكثر انتشاراً للشبكات اللاسلكية كما ذكرنا والتي تغطي تلك الطويلة والقصيرة المدى هي WPAN, WLAN, WWAN, Satellite وفيما يلي جدول يوضح أهم خصائص هذه الشبكات.

أساسيات الترددات الراديوية

لا يتطلب تطوير التطبيقات معرفة تفصيلية بالعمل الداخلي للشبكات اللاسلكية، ولكن معرفة معلومات عن آلية العمل يساعد في فهم الطريقة التي تتصرف بها التقنيات اللاسلكية المختلفة.

قبل الغوص في التقنيات اللاسلكية سنبدأ بمقدمة عن الترددات الراديوية.

الترددات الراديوية:

تنقل العديد من التقنيات اللاسلكية في شبكات WWAN, WPAN, WLAN معلوماتها باستخدام الموجات الراديوية.

لتنفيذ هذه العملية يجري تحميل البيانات على موجة راديوية نطلق عليها اسم الحامل. ويُطلق على هذه العملية اسم الترنيم (Modulation). هناك العديد من التقنيات المستخدمة في عملية الترنيم لكل منها مزاياه وعيوبه من حيث الفعالية ومتطلبات الطاقة.

تقنيات الترنيم الشائعة هي التالية:

- **تكنولوجيا الحزمة الضيقة:** تقوم أنظمة الحزمة الضيقة بإرسال واستقبال البيانات باستخدام تردد راديوي محدد. تجري المحافظة على بقاء التردد المُستخدَم ضيقاً ما أمكن. يجري تجاوز موضوع التداخل بتشغيل عدة مستخدمين على عدة ترددات وتصبح مسؤولية المُرشِّح المُستقبل استبعاد كل الإشارات عدا تلك المُرسَلة على الترددات المحددة. لتتمكن شركة ما من استخدام تقنية الحزمة الضيقة يجب أن تحصل على تصريح حكومي لحجز التردد.
- **تكنولوجيا الحزمة المنتشرة :** تقوم هذه التقنية في تصميمها بمقايضة بين فعالية عرض الحزمة من جهة والموثوقية، والاعتمادية والأمان من جهة أخرى. فهي تستهلك عرض حزمة أكبر من تلك المعتمدة على حزمة تردد ضيقة ولكنها تولد

إشارة أقوى وأسهل اكتشافاً من قبل المُستقبِلات التي تعرف المؤشرات الخاصة بالإشارة المرسلة.
تبدوالإشارة التي تستخدم تكنولوجية الحزمة المنتشرة كضجيج بالنسبة لكل المستقبلات عدا المستقبلات التي أرسلت إليها تلك الإشارة.

أساسيات الترددات الراديوية
(تقنية الحزمة المنتشرة)

هناك شكلان لهذه التقنية يمكن استخدامهما، هما تقنية قفزات التردد وتقنية التتالي المباشر

تقنية قفزات التردد FHSS : تستخدم هذه التقنية حامل ضيق الحزمة يقوم بالدوران بصورة سريعة على الترددات.
يعلم كل من المرسل والمستقبل في هذه التقنية بتشكيل الترددات المستخدمة.
الفكرة هي أنه حتى لوتم اعتراض تردد ما، تظل الترددات الأخرى متوفرة للإرسال وإلا فتتم إعادة عملية إرسال البيانات.
النتيجة تكون في نقل المعلومات بواسطة قناة منطقية وحيدة.
أما بالنسبة للمستقبلات التي لا تملك تشكيل الترددات المستخدمة يبدوالإرسال وكأنه دفقات قصيرة من الضجيج.
الحد الأقصى لمعدل الإرسال باستخدام FHSS هو بحدود 1Mbps

تقنية التتالي المباشر DSSS:
تقوم هذه التقنية بنشر الإشارة على حزمة واسعة من الترددات الراديوية بصورة متوازية. لكل (بت) يتم إرساله هناك (بت) مكرر يسمى الرُقاقة.
كلما ازداد طول هذه الرقاقة كلما كان بالإمكان معالجة واستعادة البيانات في حال تضررها.
تتطلب اليتات الطويلة أيضاً عرض حزمة أعلى.
بالنسبة للمستقبلين الذين لا يتوقعون تلقي الإشارة فإن الإشارة المُرسَلة بتقنية DSSS تبدوكضجيج ضعيف عريض الحزمة يجري رفضه.
تتطلب DSSS طاقة أكبر من FHSS ولكن يمكن رفع معدل الإرسال إلى قيمة أقصاها 2Mbps.

أساسيات الترددات الراديوية
تقنية المُضاعَفَة بتقسيم التردد المتعامد(OFDM)

يتم في هذه الطريقة إرسال المعلومات بطريقة متوازية بشكل معاكس لتقنية القفز المستخدمة في FHSS، والانتشار المستخدمة في DSSS.

يمكن من خلال هذه التقنية إرسال كمية أعلى من البيانات عبر عرض حزمة أضيق، مما يجعل التقنية OFDM مناسبة لإرسال البيانات بمعدل إرسال عالي.

تكمن المشكلة في هذه التقنية في أنها صعبة التطبيق وتتطلب طاقةً أعلى من كلا التقنيتين FHSS و DSSS.

الشبكات اللاسلكية الشخصية
WPAN

دفع الاستخدام المتزايد للتجهيزات الالكترونية في المنازل والمكاتب إلى المزيد من الحاجة إلى اتصال هذه التجهيزات ببعضها البعض بأكبر قدر من المرونة. وكان لا بد من التركيز على تطوير شبكات لتخديم عملية ربط هذه التجهيزات.

من أبرز الأمثلة عن هذه التجهيزات: الحواسب الشخصية، والحواسب المحمولة، والهواتف النقالة، والطابعات، والكثير من التجهيزات الأخرى.

كانت فكرة ربط تللك التجهيزات بواسطة كابلات مباشرة عملية مزعجة وازدادات الأمور تعقيداً حين تم إضافة المحمولية إلى تلك التجهيزات.

ظهرت الشبكات اللاسلكية الشخصية كاستجابة لتلك الحاجة وكانت أبرز الخصائص التي ميزت الشبكات WPAN هي:
- الاتصال قصير المدى
- الاستهلاك القليل للطاقة
- الكلفة المنخفضة
- شبكات شخصية صغيرة تؤمن الاتصال ضمن مساحة العمل الشخصية.

المعايير القياسية المستخدمة في شبكات
WPAN

استُخدِمت العديد من المعايير القياسية للشبكات اللاسلكية الشخصية، بحيث امتلك كل منها نقاط قوة ونقاط ضعف.
سنستعرض في هذا الجزء من الجلسة بعض هذه المعايير وأغراض استخدامها:

معيار IrDA:
يعود هذا الاختصار إلى المؤسسة العالمية التي قامت بوضع معايير الاتصال التبادلي منخفض الكلفة باستخدام الأشعة تحت الحمراء.
وضع المعيار IrDA مجموعة من البروتوكولات لدعم التجهيزات، والحواسب، وأجهزة الاتصال.
كان الغرض من هذه البروتوكولات توفير نقل بيانات لاسلكي من نقطة إلى نقطة عبر خط نظر ولمسافة قصيرة وبسرعة عالية.
تستخدم برتوكولات IrDA الـــIrDA DATA كآلية تسليم للمعلومات والـــ IrDA CONTROL كآلية تحكم.
يكمن الغرض الأساسي من IrDA في تقديم بديل عن الوصل بالكابل. إذ يكفي بكل بساطة مقابلة بوابات الأشعة تحت الحمراء لجهازين

للتمكن من إرسال واستقبال البيانات بينهما. وفيما يلي بعض أهم الخصائص هذا النوع من الاتصال:

- يصل مجال الاتصال حتى 1 متر وإن كان بالإمكان في بعض الأحيان الوصول حتى 2 متر
- تقدم هذه التقنية خياراً لتوفير الطاقة في حالات الاتصال ضمن مجال حتى 20 سنتيمتر. حيث يتطلب هذا الخيار طاقة أقل بعشر أضعاف من حالة الاستثمار بالطاقة القصوى.
- الاتصال ثنائي الاتجاه.
- سرعة إرسال البيانات تتراوح بين 9600 bps وحتى 4 Mbps.
- تكون التقنية منخفضة الكلفة حيث يمكن تضمين هذه التقنية ضمن جهاز بكلفة لا تتجاوز دولاراً واحداً.

ولكن، بالرغم مما يقدمه هذا المعيار من ميزات، تبقى هناك الكثير من العوائق التي لا تُمكِّن هذه التقنية من تغطية معظم احتياجات الاتصالات الشخصية.
فمن الممكن أن تكون فكرة استخدام الأشعة تحت الحمراء مناسبة في حالة تبادل معلومات اتصال ما بين شخصين يحملان جهازين محمولين، ولكن الأمور تصبح أكثر تعقيداً عند الحاجة إلى الاتصال بطابعة مثلاً، والتي قد لا نستطيع الحصول على خط نظر مباشر بينها وبين التجهيزات التي تحاول الاتصال بها في بيئة مكتب أوفي البيئة المنزلية.

المعايير القياسية المستخدمة في شبكات WPAN

المعيار BlueTooth

بلوتوث هومعيار يساعد في الاتصال اللاسلكي بين الحواسب المحمولة، والهواتف النقّالة، وبعض التجهيزات المحمولة الأخرى.

بصورة مغايرة لما لحظناه في حالة الأشعة تحت الحمراء، لا تتطلب التجهيزات المزودة بتقنية بلوتوث خط نظر بين التجهيزات المتصلة، وتستطيع تأمين الاتصال عبر الحواجز.

تكون مسافة الاتصال في الحالة العادية 10 متر ولكن يمكن الوصول حتى 100 متر باستخدام مضخمات إشارة.

يستخدم معيار بلوتوث الطيف 2.4-GHZ غير المُرخَّص للاتصال، كما يمتلك معدل نقل يصل إلى 720Kbps ومن المتوقع في المستقبل القريب أن يتم رفع معدل النقل ليصل إلى 10Mbps.

تم تطوير هذا المعيار من قبل شركات كبرى حيث بدأت به شركة Ericsson، ثم تعاونت معها شركات مثل Nokia،INTEL، Toshiba وIBM لتشكل ما يسمى بـــ SIG،
(Bluetooth Special Interest group) ثم انضمت العديد من هذه الشركات إلى هذه المجموعة مثل Microsoft , Motorola, Lucent , 3COM وأكثر من ألفي شركة أخرى.

تبقى التوقعات المرتبطة بهذا المعيار كبيرة، ولكن بلوتوث ما يزال حتى الآن يعمل وبشكل أساسي كبديل عن الاتصال بالكابل.

يقدم بلوتوث آلية اكتشاف تلقائي حيث يمكن للتجهيزات المزودة بهذه التقنية اكتشاف جميع الأجهزة الواقعة ضمن مجال عملها، بحيث يجري بعدها تأسيس الاتصال بين تلك الأجهزة.

هناك بعض التخوف من التحميل الزائد للطيف 2.4GHZ مع استخدام المزيد من التجهيزات التي التي تستعمل تقنية بلوتوث. لتجاوز هذه المشكلة تُعرِّف محددات المعيار بلوتوث ثلاث أنماط للتجهيزات:

التجهيزات القابلة للاكتشاف بشكل عام : يسمح هذا النمط بأن يتم اكتشاف الجهاز الذي يستخدم هذه التقنية من أي جهاز آخر ضمن المجال.

التجهيزات القابلة للاكتشاف بشكل محدود: يمكن فقط لتجهيزات محددة ومُعَّرفة مُسبقاً أن تكتشف بعضها البعض، يُستَخدَم هذا النمط بصورة أساسية عند وجود مجموعة من تجهيزات بلوتوث التي نرغب أن تتعرف على بعضها البعض أتوماتيكياً.

التجهيزات غير القابلة للاكتشاف: يجعل هذا النمط الجهاز غير مرئي وغير قابل للاكتشاف من قبل التجهيزات الأخرى.

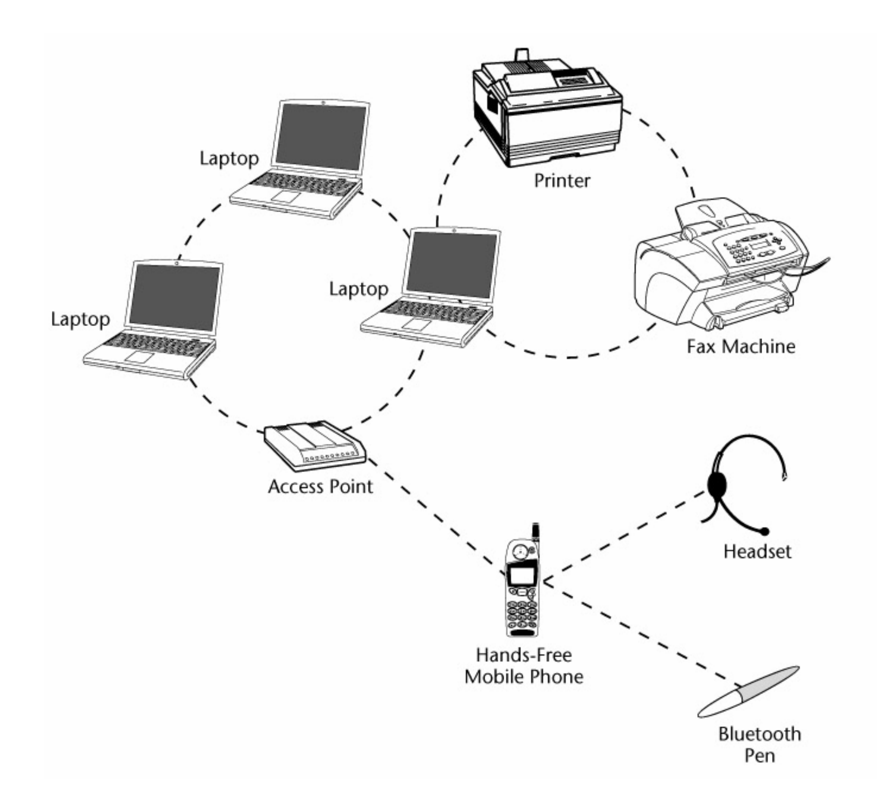
المعايير القياسية المستخدمة في شبكات WPAN

المعيار BlueTooth

ما أن يتم الاتصال بين جهازين باستخدام بلوتوث يتم تشكيل شبكة PICONET يمكن أن تتــألف من 8 أجهزة على الأكثر بحيث يمكن لأي من تلك الأجهزة الاتصال مباشرةً مع الآخر.

لتأليف شبكة مكونة من أكثر من ثمان أجهزة لا بد من استخدام أكثر من PICONET تُدمج مع بعضها بما يسمى الشبكة المُبعثرة بحيث لا يمكن لجميع التجهيزات في الشبكة التخاطب وإنما فقط تلك الموجودة في كل Piconet تستطيع التخاطب مع التجهيزات الموجودة في نفس الـــ Piconet.

في الشكل التالي يوجد شبكة مبعثرة واحدة مؤلفة من خمس Piconet



ما أن يتم الاتصال بين جهازين باستخدام بلوتوث يتم تشكيل شبكة PICONET يمكن أن تتــألف من 8 أجهزة على الأكثر بحيث يمكن لأي من تلك الأجهزة الاتصال مباشرةً مع الآخر.

لتأليف شبكة مكونة من أكثر من ثمان أجهزة لا بد من استخدام أكثر من PICONET تُدمج مع بعضها بما يسمى الشبكة المُبعثرة بحيث لا يمكن لجميع التجهيزات في الشبكة التخاطب وإنما فقط تلك الموجودة في كل Piconet تستطيع التخاطب مع التجهيزات الموجودة في نفس الـــ Piconet.

المعايير القياسية المستخدمة في شبكات
WPAN
المعيار BlueTooth

التشكيلات الخاصة بالمعيار BlueTooth

عند تطوير المعيار الخاص ببلوتوث ولضمان التعامل بين التجهيزات التي تزود هذه التقنية قامت SIG بتحديد مجموعة من التشكيلات عددها 13 لضمان العمل المشترك لهذه التجهيزات وفق أساس واحد.

تم تصميم كل تشكيل ليقوم بأداء مهمة محددة.

تم تصميم أربعة تشكيلات رئيسية لتكون اللبنات الأساسية التي ستعتمد عليها التشكيلات الأخرى.

التشكيلات التسعة الباقية هي تشكيلات الاستخدام وتعبر كل منها عن شكل من أشكال العمليات التي يمكن إجراءها بواسطة هذه التقنية.

لم يتم تصميم التشكيلات المذكورة لتحديد طريقة واحدة لاستخدام تقنية بلوتوث ولكنها صممت فقط كمعيار قياسي لضمان عمل التجهيزات مع بعضها البعض أي يمكن لأي مصنع تقديم الآليات التي يرغب بها على أن تكون مؤسسة على هذا المعيار لضمان عمل المتبادل للجهاز مع الأجهزة الأخرى التي تستخدم تقنية بلوتوث.

لمزيد من المعلومات حول هذه التشكيلات يمكن زيارة الموقع www.bluetooth.com.

الأمان في المعيار BlueTooth:

لما كانت أحد أولويات تقنية بلوتوث أن تكون بديلاً عن استخدام الكابلات فلا بد لها أن تقدم سوية أمان مساوية على الأقل لما هومتوفر في حالة الكابلات.

تحدد خصائص الأمان في تقنية بلوتوث في طبقة الربط (من طبقات OSI). أما على مستوى التطبيق فلم يتم تحديد أية آلية حيث تم ترك هذه السوية للمطور لتحديد الآلية الأنسب حسب التطبيق المستخدم.

يمكن للاتصال باستخدام بلوتوث أن يكون مشفراً كذلك يقدم آلية تحقق مبيتة ضمن الجهاز.

يمكن أن يحدد المستخدم درجة التشفير ويمكن أن يستخدم مفتاح بطول يتراوح بين 8 إلى 128 بيت.

يمكن هذا المستخدم وحسب درجة الأمان المطلوبة تحديد التوازن الأفضل بين سرعة الاتصال ومستوى الأمان.

المعايير القياسية المستخدمة في شبكات WPAN المعيار 802.15

المعيار 802.15:

هومجموعة من الخصائص التي حددها معهد (IEEE) لتطوير المعايير للشبكات اللاسلكية قصيرة المدى أوالشبكات اللاسلكية الشخصية.

عند البدء بوضع هذا المعيار تمت الاستفادة من المواصفات المُستخدَمة في تقنية بلوتوث واستخدام جزء منها كأساس لمعيار 802.15.

يهدف هذا المعيار بصورة أساسية إلى تقييس التحكم بالوصول إلى الوسط (MAC) كذلك الطبقة الفيزيائية من تقنية بلوتوث ويتطرق إلى مواضيع مثل التعامل المتبادل على الشبكة. لهذا الغرض جرى إنشاء أربع مجموعات مهام للعمل على المكونات المختلفة لهذا المعيار هي:

- مجموعة مهام WPAN/BlueTooth
- مجموعة مهام Coexistence Mechanisms
- مجموعة مهام WPAN عالي المعدل
- مجموعة مهام WPAN منخفض المعدل – بعمر بطارية طويل.

الشبكات اللاسلكية المحلية
WLAN

تعد حلول الشبكات اللاسلكية المحلية من أكثر القطاعات سرعةً في التطور في مجال صناعة الاتصالات.
أهم استخدامات هذه الشبكات كانت توفير الكلف وتجنب مد الكابلات بالإضافة إلى تقديم اتصال انترنت بسرعة عالية للمستخدمين.
وما تزال الكثير من الاستخدامات الهامة لهذه الشبكات تظهر كل يوم.

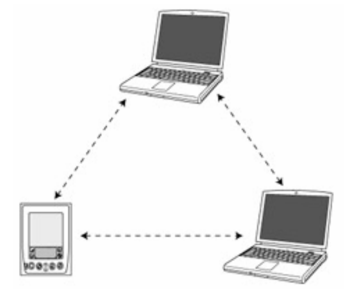
عند تطوير تجهيزات هذا النوع من الشبكات كان لا بد من تحديد مجموعة من المُحددات الأساسيات التي تشمل:

- **التغطية** : عادة ما تغطي هذه التجهيزات مساحة بنصف قطر يتراوح بين 50 و150 متراً.
- **سرعة نقل البيانات:** من 1 Mbps إلى 54 Mbps
- **التداخل:** يجب دراسة التداخل مع شبكات من أنواع أخرى.
- **استهلاك الطاقة:** يختلف استهلاك الطاقة من قبل موائمات الشبكة اللاسلكية بحسب المعيار الذي تستخدمه.
- **السعر:** يرتبط بالمتطلبات والمعيار المستخدم.

ترتيب شبكات WLAN

يتدرج ترتيب شبكات WLAN من تلك البسيطة جداً حتى شديدة التعقيد.

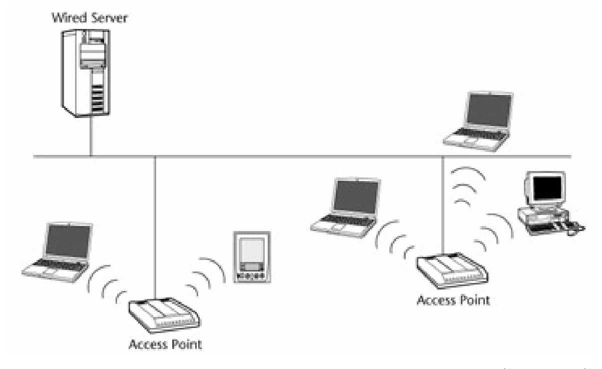
أبسط شكل لترتيب هذه الشبكات هونمط النظير إلى النظير حيث يمكن في هذا الترتيب اتصال موائمي شبكة أوأكثر ببعضهما البعض. عادة ما يسمى هذا الترتيب **بالاتصال الخاص** (ad hoc). لا يتطلب هذا الترتيب وجود نقطة ولوج لاسلكية حيث يتم الاتصال بصورة مباشرة ودون الحاجة إلى المرور بعقدة مركزية أي لا يتطلب إدارة أوإعداد مسبق.



الترتيب الخاص (Ad hoc)

أما الترتيب البنيوي المؤلف من نقطة ولوج لاسلكية الاتصال بالشبكة السلكية، فيؤمن، بالإضافة –إلى ميزة مُشاركة عدة تجهيزات متصلة مع نقطة ولوج لاسلكية– اتصالاً سريعاً وواسعاً مرتبطاً بهذه النقطة.

يُفضّل في هذا الترتيب أن تكون مساحات تغطية نقاط الولوج متقاطعة لتأمين استمرارية في الاتصال دون أي انقطاع على الحدود بين مساحات التغطية. يمكن في هذا النوع من الترتيب استخدام تجهيزات نقاط التوسيع (وهي عبارة عن مقويات إشارة تقوم فقط بتلقي الإشارة وإعادة بثها) التي لا تتطلب الاتصال السلكي بالشبكة، لذلك تُستخدَم لتوسيع الشبكة للاسلكية في المناطق البعيدة نسبياً. ويُعتَبَر استخدام الهوائيات الموجهة، والتي تسمح بإرسال الإشارة باتجاه محدد لاستقبالها اعتماداً على هوائي مقابل بهدف إيصال الإشارة إلى نقطة ولوج أخرى، أحد أساليب توسيع امتداد الشبكة أيضاً.



الترتيب البنيوي

معايير شبكات WLAN

تبنت العديد من الهيئات الواضعة للمعايير وضع معايير خاصة بشبكات WLAN من أهم هذه الهيئات IEEE وETSI وقام بالتسويق لهذه المعايير الاتحاد HomeRF

هناك ثلاث معايير أساسية تستحق الاهتمام الأكبر :

المعيار 802.11:

يعد أول معيار تم تعريفه لشبكات WLAN، يستخدم هذا المعيار نفس البروتوكولات الخاصة بالتبديل والمُستَخدَمة في شبكات Ethernet السلكية. ولكنه يسمح بالاتصال اللاسلكي باستخدام التردد الراديوي 2.4Ghz

يدعم هذا المعيار تقنيتي التعديل FHSS وDSSS.

لم تعد التجهيزات الخاصة بهذا المعيار منتشرة بشكل كبير نتيجة حلول المنتجات التي تدعم المعايير 802.11a و802.11b بدلاً عنها.

المعيار 802.11b/Wi-Fi

يعد هذا المعيار من أكثر المعايير انتشاراً في عائلة 802.11x. يستخدم هذا المعيار الآلية DSSS باستخدام الطيف 2.4Ghz. يمكن للمعيار 802.11b أن يصل إلى السرعة 11Mbps ولمسافة تصل إلى 100 متر.

هناك مقايضة تتم بين المسافة ومعدل النقل وتتدرج من 1Mbps لمسافة 100 متر إلى 11Mbps لمسافة 30 متر.

قام اتحاد (WECA) بإعداد شهادة باسم WiFi تتضمن إمكانية عمل منتج ما يعمل باستخدام المعيار 802.11b مع منتج آخر حاصل

على هذه الشهادة.

معايير شبكات WLAN

المعيار 802.11a:

يعد المعيار 802.11a بديلاً يوفر سرعة أكبر من 802.11b ويستخدم التردد 5Ghz وبسرعة تصل حتى 54Mbps.
بعكس المعيار 802.11 و802.11b، يستخدم المعيار 802.11a الترنيم من نوع OFDM.

إن استخدام تردد مختلف وآلية ترنيم مختلفة يؤدي إلى عدم توافق شبكات المعيار 802.11a مع شبكات المعيار 802.11b.

حمل هذا المعيار العديد من الميزات منها رفع سرعة نقل البيانات Mbps54 والتقليل من التداخل نظراً لاستخدام تردد أقل ازدحاماً.
ولكن في نفس الوقت انخفض المجال الأقصى الذي تغطيه نقطة الولوج من 100 متر في حالة المعيار 802.11b إلى 50 أوحتى 25 متر في حالة المعيار 802.11a.

كما يبرز أيضاً الفرق في استهلاك الطاقة بين المعيارين فالمعيار 802.11a يستخدم كما ذكرنا تقنية OFDM والتي تستهلك طاقة أكبر من التقنية DSSS المُعتمدة في المعيار 802.11b.

معايير شبكات WLAN

المعيار 802.11g:

يقدم هذا المعيار الاتصال اللاسلكي بسرعة عالية على التردد 2.4Ghz مع المحافظة على التوافقية مع المعيار 802.11b.
يتم الوصول إلى هذه الميزات عن طريق طبقتين الأولى تعتمد المعيار 802.11g تعمل على التردد 2.4Ghz المُستخدم في المعيار 802.11b وباستخدام نفس نوع التعديل DSSS حيث يمكن الحصول على سرعة تصل حتى 11Mbps.
أما الطبقة الثانية فتعتمد تعديل أكثر فعالية هو OFDM باستخدام نفس التردد 2.4Mhz للوصول إلى معدل نقل للبيانات 54Mbps.

عملياً تستطيع بطاقات الشبكة ونقاط الولوج العاملة باستخدام المعيار 802.11b العمل مع بطاقات الشبكة ونقاط الولوج العاملة باستخدام المعيار 802.11g لكن سيُخفِّض أي مكون يعمل بالمعيار 802.11b معدل نقل البيانات إلى قيمة أقصاها 11Mbps. لذا يجب أن تكون المكونات جميعها عاملة على المعيار 802.11g للوصول إلى سرعة 54Mbps.
تتشابه النواحي الأخرى مثل تلك الخاصة بمساحة التغطية للشبكة بين المعيارين 802.11b و802.11g.

معايير 802.11 الأخرى:

حددت IEEE مجموعة من المعايير الأخرى الخاصة بإجراءات مختلفة أهمها:
المعيار IEEE 802.11e. المخصص لتأمين محددات جودة الخدمة اللازمة لتأمين الحصول على اتصالات صوتية ذات اعتمادية على الأنظمة المتوافقة مع 802.11b.
المعيار 802.11f. المخصص لتطوير بروتوكول الولوج المتبادل بين نقاط الولوج.
المعيار 802.11h. المخصص لتحسين الطبقة الفيزيائية السريعة الخاصة بالمعيار 802.11a العامل على التردد 5Ghz لجعل تجهيزات المعيار 802.11a تخضع لمتطلبات الأنظمة الأوروبية.

المعيار 802.11i المخصص لتحسين الطبقة MAC للمعيار 802.11 وذلك لزيادة السرية وتحسين آليات التحقق من الهوية.

معايير شبكات WLAN

معيار HomeRF:

كما يوحي اسم هذا المعيار فهوخاص بالشبكات المحلية اللاسكية الخاصة بالتطبيقات المنزلية.

تم تأسيس هذا المعيار على المعيار 802.11 باستخدام FHSS ولكن تم القيام ببعض التعديلات لتغطية احتياجات أغلب المستهلكين.

يستخدم هذا المعيار البرتوكول SWAP (Shared Wireless Access Protocol) الذي يدعم الاتصال الصوتي بجودة عالية مع دعم للمعيار الخاص بالهواتف اللاسلكية DECT، ويسمح باستخدام البنية التحتية اللاسلكية للهاتف وللاتصال الشبكي مع تأمين عدد من الميزات الهاتفية المتقدمة مثل انتظار المكالمات وتحويلها إضافة إلى تخصيص الرنات...إلخ.

تم التركيز في هذا المعيار على عامل السعر أكثر من سرعة النقل حيث يمكن لشبكات هذا المعيار تغطية حتى 50 متراً والوصول إلى سرعة أقصاها 10Mbps.

يستخدم هذا المعيار أيضاً التردد 2.4Ghz.

المعيار HIPERLAN/1 وHIPERLAN/2:

طرح معهد المعايير الأوروبي للاتصالات (ETSI) المعيار HIPERLAN أي الشبكة المحلية الرادوية عالية الأداء.

تم طرح نسختين من هذا المعيار الأولى اعتمدت الأولى التردد 5Ghz ولكنها لم تصل إلى حيز التطبيق ولم يتم إنتاج أي تجهيزات تحت هذه النسخة. أما النسخة الثانية من هذا المعيار فهي أيضاً تستخدم التردد 5Ghz ولكن باستخدام التعديل OFDM حيث يمكن للتجهيزات وفق هذا المعيار تغطية مسافة بطول 150 متراً وبسرعة أقصاها 54Mbps، كما ركز هذا المعيار على تحسين جودة الخدمة للاتصالات متعدد الوسائط، وعمل على تحسين استهلاك الطاقة للتجهيزات المحمولة التي تعتمد هذا المعيار.

لم يُطبَق هذا المعيار حتى الآن في التجهيزات ولكنه يعتبر أحد المعايير التي سيكون لها دور كبير في المستقبل القريب.

شبكات الأقمار الصناعية

تم التفكير في أنظمة الأقمار الصناعية للاتصال اللاسلكي للصوت والبيانات في بداية التسعينات حيث كان الهدف تقديم شبكة ذات تغطية كبيرة تشمل مجموعة من البلدان أوحتى الكوكب بأكمله دون الحاجة إلى أكثر من هاتف نقال أوعمليات الانتقال من نقطة ولوج إلى أخرى.

لم تكن معايير WWAN في هذه المرحلة واضحة وكان أغلب هذه الشبكات تماثلياً، وكان من الواضح أن الهواتف النقالة المعتمدة على الأقمار الصناعية ستصبح حقيقة تفرض نفسها كمعيار للاتصال ضمن مساحات واسعة.

لعب عنصر السعر دوراً كبيراً في الحد من انتشار الهواتف المعتمدة على الأقمار الصناعية بسبب توفر الحل الأقل كلفة والمتمثل في الهواتف الخليوية. رغم هذا أثبت العديد من تطبيقات شبكات الأقمار الصناعية فعالية في الحالات التالية:

- **في المواقع الثابتة** : عندما تكون هناك ضرورة للاتصال اللاسلكي من موقع ليس مُخدّماً بأي نوع آخر من الشبكات كما هي

الحال في شركات النفط والغاز التي تتوضع مقرات عملها في أماكن بعيدة معزولة.

- **الاتصالات المحمولة** : حيث يمكن للأشخاص الاتصال صوتياً أو إرسال البيانات بغض النظر عن موقعهم كما في حالة فرق الإنقاذ، الأبحاث،والقوى العسكرية.
- **العربات** : وهي أحد أهم الحالات التي يمكن الاستفادة فيها من الاتصال باستخدام الأقمار الصناعية حيث يمكن للعاملين في مجال الأخبار والتوصيل الاستفادة من هذا النوع من الاتصال لضمان الاتصال بغض النظر عن موقعهم.

سرعة نقل البيانات لهذا النوع من الاتصال الشبكي بين 2.4Kbps و2Mbps وذلك بحسب الحل المستخدم.

الشبكات اللاسلكية العريضة
WWAN

تم استعمال هذا النوع من الشبكات لأول مرة في بداية الثمانينات حيث كان استخدام هذه الخدمة يتطلب التسجيل على الخدمة وتسديد مبلغ يتحدد بالدقائق التي تم استخدام هذه الشبكة خلالها أومؤخراً كمية البيانات التي تم نقلها عبر هذه الشبكة.
لا تعمل شبكات WWAN وبعكس شبكات WPAN وWLAN على ترددات غير مرخصة أي أنه يتوجب تسديد مبلغ مالي لحجز تردد ما.
فيما يلي سنستعرض بعض الأفكار الضرورية لفهم عمل الأنواع المختلفة لشبكات WWAN.

الإشارات التماثلية والإشارات الرقمية:

الإشارة التماثلية : يمثل عادة هذا النوع من الإشارات بموجة جيبية.
استخدم هذا النوع من الإشارات في الجيل الأول من الشبكات اللاسلكية العريضة وذلك للاتصالات الصوتية.
الإشارة الرقمية : وهي عبارة عن سلسلة من القيم 0 و1 التي يتم إرسالها وذلك للوصول إلى أقرب صيغة يمكن للحاسب التعامل معها وهي التنسيق الثنائي.
تم استخدام هذا النوع من الإشارات في شبكات الجيل الثاني اللاسلكية.
تعتبر الإشارات الرقمية أكثر فعالية وأماناً إضافة إلى تأمين جودة أعلى والعديد من المميزات الأخرى.

مبدلات الدارة ومبدلات الحزم:

هناك طريقتان أساسيتان تم اعتمادهما لعملية التبديل
مبدلات الدارة : تقوم الشبكات التي تعتمد هذا النوع من التبديل بتأسيس اتصال فيزيائي بين طرفين، حيث يستمر حجز دارة لهذا الاتصال مع عدم إمكانية استخدام هذه الدارة لتخديم أي اتصال آخر حتى انتهاء الاتصال الأول.
مشكلة هذا النوع من المبدلات الحاجة إلى عدد دارات مساولعدد الاتصالات المراد تأسيسها.
مبدلات الحزم:تعمل مبدلات الدارة بشكل جيد حين تكون كمية البيانات التي يتم إرسالها ثابتة كما هي الحال في الاتصالات الصوتية ولكنها تصبح غير فعالة في العديد من أنواع الاتصالات المخصصة لنقل البيانات، حيث يتم طلب المعلومات بدفقات كما هي الحال في استعراض صفحات الانترنت.
في هذه التقنية يمكن لأكثر من مستخدم الاشتراك في نفس الاتصال مما يساعد على الاستثمار الأمثلي للاتصال.

تتم هذه العملية بتقسيم البيانات إلى أجزاء تدعى حزم وإرسالها مرفقة بعنوان المرسل إليه، حيث تتم إعادة توجيهها عبر الاتصالات المختلفة حتى الوصول إلى هدفها.

الشبكات اللاسلكية العريضة
WWAN

تم تصنيف الشبكات اللاسلكية العريضة حسب تطورها إلى أربع مراحل هي الجيل الأول، الجيل الثاني،الجيل الثاني والنصف والجيل الثالث

شبكات الجيل الأول 1G:

ظهر هذا النوع من الشبكات في نهاية السبعينات وبداية الثمانينات.غالباً ما استخدمت شبكات هذا الجيل في الاتصالات الصوتية.

كان من أهم العيوب لهذه الشبكات جودة الاتصال المنخفضة والتداخلات المستمرة

شبكات الجيل الثاني 2G:

استخدمت شبكات الجيل الثاني التقنية الرقمية التي ساعدت بشكل كبير في تقديم اتصالات صوتية بجودة عالية إضافة إلى خدمات البيانات الأساسية.

سمحت هذه الشبكات لعدد أكبر من المستخدمين بالاتصال على نفس مجال الحزمة وبالتالي رفع كفاءة الشبكة بصورة عامة إضافة إلى رفع سوية الأمان على الشبكة.

تدعم معظم شبكات الجيل الثاني نقل البيانات

أهم شبكات الجيل الثاني المستخدمة حالياً:

شبكات AMPS (خدمة الهواتف المحمولة المتقدمة وتسمى أحياناً ب TDMA).

شبكات CDMA IS-95 أوما يسمى بشبكات الوصول المتعدد بالتقسيم حسب الشفرة المستخدمة.

شبكات GSM (النظام العام للاتصالات المحمولة) ويعد من الأنظمة الأكثر شيوعاً.

شبكات PDC (الخليوي الشخصي الرقمي) المستخدم بشكل واسع في اليابان.

شبكات الجيل الثاني والنصف (2.5G)

كان هذا الجيل خطوة إضافية باتجاه شبكات الجيل الثالث. من أبرز ما ميز هذا الجيل من الشبكات التحول من استخدام مبدلات الدارة إلى التبديل بالحزمة مما مكن الوصول إلى سرعة 144Kbps أي ما يتجاوز عشرة أضعاف ما هي عليه في شبكات الجيل الثاني.

من أهم الشبكات في هذا الجيل :

شبكة CDMA 2000 1x

شبكة GPRS

شبكات الجيل الثالث:

بدأ تطوير شبكات الجيل الثالث ضمن رؤية لتوحيد المعيار المستخدم لهذه الشبكات في جميع البلدان، ونظراً لاعتماد الكثير من البلدان تقنيات مختلفة وللمحافظة على التوافقية مع الأنظمة الأقدم ظهر أنه من شبه المستحيل الوصول إلى معيار يغطي هذه الأنواع كلها.

لذلك تم الاعتماد على ثلاث فروع أساسية لأنظمة شبكات الجيل الثالث وهي WCDMA، CDMA2000، EDGE

القسم الرابع:

بنيان التطبيقات المحمولة وتقنية التراسل

الكلمات المفتاحية:

محمول، لاسلكي، تقنية، تجهيزات، تطبيق، تراسل

ملخص:

تعتمد التطبيقات المحمولة عدة أنماط من البنيان، سنحاول في هذه الجلسة تغطية أهم هذه الأنماط التطبيقات المحمولة، ونقاط القوة والضعف في كل منها إضافةً إلى أمثلةٍ عن التطبيقات التي تعتمد هذه الأنماط.

أهداف تعليمية:

يتعرف الطالب في هذا الفصل على:

- البنيانات المختلفة للتطبيقات المحمولة
- محاسن ومساوئ كل منها
- استعراض مجموعة من التطبيقات التي تعتمد كل من هذه البنيانات
- تقنية التراسل واستخدامها.

بنيان التطبيقات المحمولة

تتدخل الكثير من العوامل في تحديد طبيعة الحلول المحمولة ونجاعتها. من أهم تلك العوامل بنيان تلك التطبيقات.

سنتحدث في هذه الجلسة عن ثلاث نماذج لبنيان التطبيقات المحمولة وهي:

- الانترنت اللاسلكية
- الزبون الذكي
- التراسل.

العوامل التي تؤثر في اختيار بنيان التطبيقات

مستخدمو التطبيق:

يؤثر نوع المستخدمين بشكل كبير على البنيان المُستخدم في التطبيق حيث لا بد من تحديد من هم هؤلاء المستخدمون، ما هي مهارتهم التقنية وما هي سيناريوهات الاستخدام التي قد يتبعوها.

نوع التجهيزات:

يجب التفكير هنا بإمكانية دعم التجهيزات المتوفرة للتطبيق، وما هي التجهيزات الأنسب للتعامل معه، وهل توفر التجهيزات إمكانية الاتصال اللاسلكي مباشرةً أم أنها تحتاج للربط مع جهاز آخر ليؤمن لها الاتصال.

تُضاف إلى ماسبق عوامل تتعلق بقدرة الجهاز على تأدية أكثر من عمل في وقت واحد، كالاتصال بالانترنت مثلاً وإرسال رسائل SMS في نفس الوقت.

الاتصال بالمؤسسة:

- ما هي الآلية التي ستتم بها عملية الاتصال مع المؤسسة؟ وهل هي لاسلكية أم سلكية (كحال اتصال يعتمد وصلة USB)؟
- في حالة الاتصال اللاسلكي، ماهونوع الشبكة المُستخدمة (WLAN, WPAN, WWAN أواتصال عبر الأقمار الصناعية)؟
- ما هو نوع الاتصال المُستخدَم وما مدى تأثيره على كمية البيانات التي يجب إرسالها من التطبيق المحمول إلى مخدم المؤسسة؟

بيانات المؤسسة:

- ما هوحجم البيانات الواجب توفره لمستخدم التجهيز المحمول؟
- أين يجب أن تتوضع البيات، على الطرفية المحمولة للزبون أم على مخدم المؤسسة؟
- هل من المجدي تحميل البيانات في الزمن الحقيقي عبر الاتصال اللاسلكي أم أن هناك ضرورة لعملية تخزين للبيانات على

طرفية الزبون؟

- ما هي الديمومة المطلوبة للبيانات؟ وهل ستحتاج إلى التحديث بشكل مستمر كما هي الحال في معلومات أسعار الأسهم والبورصة؟ أم تكفي عملية تحديث يومية كما هي الحال في المخازن؟
- هل يمتلك كل مستخدم جهاز وحيد للوصول إلى التطبيق أم عدة أجهزة؟
- هل يمكن للمستخدمين الاشتراك باستخدم نفس الجهاز دون المخاطرة بخلط بياناتهم؟
- هل سيكون هناك تضارب لدى تحديث البيانات للمؤسسة من الطرفيات المختلفة؟

التكامل مع المؤسسة:

- هل يوفر التطبيق إمكانية التكامل بين الزبون والمخدم في المؤسسة عبر واجهات برمجية خاصة؟

إخطار المستخدم:

- هل يحتاج المستخدم إلى استقبال تنبيهات خاصة؟
- وفي حال كان المطلوب هو إرسال إشارة إلى المستخدم فهل يمكن استخدام أجهزة الهاتف الخليوي العادية وأجهزة التنبيه لهذا الغرض؟
- وما الذي سيحدث في حال كان الجهاز المحمول للمستخدم مغلقاً؟

الأمان:

- كيف يمكن حماية البيانات الحساسة أثناء إرسالها أوعلى الجهاز؟ وماهي سوية الأمان وآلية تطبيقها ومكان تطبيقها (مخدم،زبون)؟
- هل تعتبر الفيروسات أحد الأخطار الواجب الاحتياط لها؟ وماذا عن عملية سرقة الجهاز أوضياعه؟

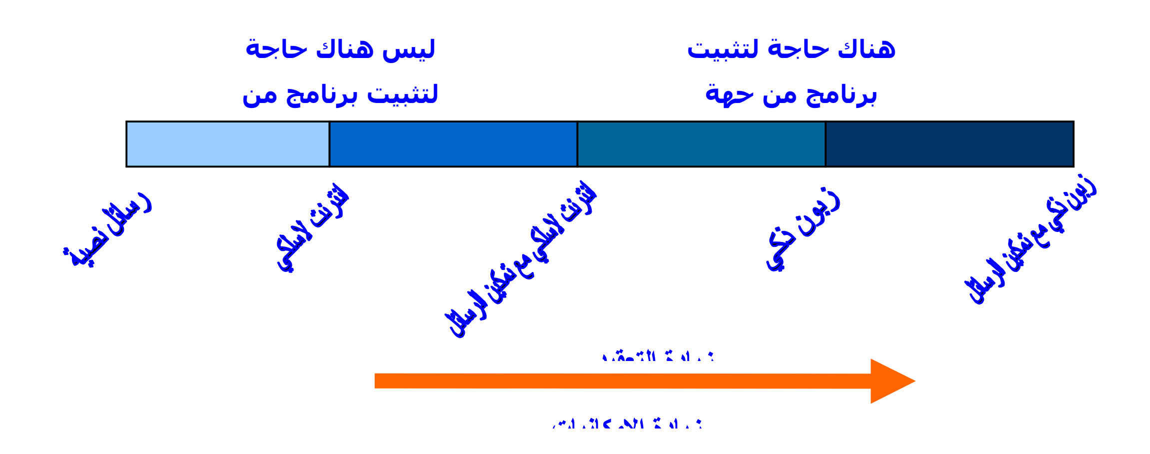
عمر البطارية:

- هل يعتبر استهلاك الطاقة أحد الأولويات؟
- ما مدى مدى استهلاك الاتصال اللاسلكي للطاقة؟
- هل تتوفر بطارية احتياطية؟ وهل يمكن أن يتم شحنها أثناء العمل؟

بنيان التطبيقات

نادراً ما تتطابق الحاجات التي يقدمها البنيان المتوفر مع جميع احتياجات التطبيقات. لذلك سيكون هناك ضرورة للمقايضة بين مختلف المزايا للوصول إلى التركيبة الأكثر مناسبة.

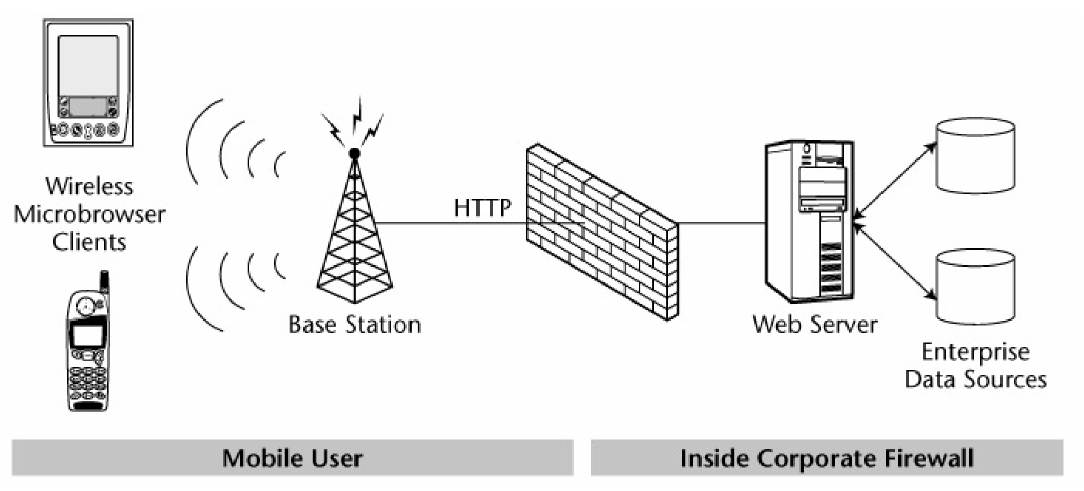
يوضح الشكل التالي العلاقة بين تعقيد الحل، وبين زيادة الإمكانيات وضرورة تثبيت برنامج على طرفية الزبون.



سنقوم فيما يلي بالاطلاع على معماريات مختلفة وذلك بغرض معرفة الفوائد التي يمكن أن نجنيها على مستوى التطبيق لهذ المعماريات هي :

1- بنيان الانترنت اللاسلكية
2- بنيان الزبون الذكي
3- بنيان تطبيقات التراسل.

بنيان الانترنت اللاسلكية



يتشابه هذا البنيان مع بنيان السلكية مع فارق يكمن في كون آلية الاتصال مع المستخدم النهائي لاسلكية. يتركز منطق العمل والبيانات في هذا البنيان عند المخدم، أما من طرف الزبون، فيجب أن يحتوي الجهاز المحمول على مستعرض أوما يسمى أحياناً مستعرض صغري دون الحاجة لتثبيت برامج أخرى من طرف الزبون. لذلك تدعى أحياناً هذه التطبيقات بتطبيقات الزبون الرقيق.

العناصر الأساسية لهذا البنيان:

المستعرض الصغري: يقوم المستعرض بإرسال طلب يحتوي عنوان المصدر القياسي URL إلى المخدم. تأتي الاستجابة من المخدم على شكل لغة تأشير حيث يقوم المستعرض بتفسيرها وإظهار المحتوى على شاشة الجهاز المحمول.

في حال كانت التقنية المحمولة لا تستخدم بروتوكول IP سيكون هناك حاجة إلى استخدام عبارة لتحويل الطلبات إلى طلبات HTTP يتم توجيهها إلى مخدم الوب.

مخدم الوب اللاسلكي: تتلخص مهمة مخدم الوب هذا في استلام طلبات HTTP وإرسال الرد المناسب. لضمان استجابة مناسبة للمستعرضات المختلفة، ونظراً لحاجة هذه المستعرضات إلى استخدام لغات تأشير مختلفة، يجري تثبيت إطار عمل خاص على المخدم يساعد في تشكيل استجابة المخدم حسب المستعرض الذي قام بإرسال الطلب.

مصدر البيانات في المؤسسة: يقوم مخدم الوب اللاسلكي بالوصول إلى مصادر البيانات في المؤسسة حيث تعتمد آلية الوصول ومصادر البيانات المعتمدة على عوامل تتعلق بحساسية البيانات وحجم الوصول ووجود آليات وصول بديلة.

محاسن ومساوئ بنيان الانترنت اللاسلكية

المحاسن:

- **تكون الحاجة إلى تثبيت برمجيات على طرف الزبون معدومة أوبالحد الأدنى.**
- **توسيع الإطار الذي يستخدم الانترنت** حيث تؤمن هذا البنيان امتداد طبيعي للتطبيقات التي تعمل على الانترنت السلكية
- **واجهة المستخدم المألوفة:** حيث أن معظم المستخدمين معتادين على استخدام واجهة المستعرض في تطبيقاتهم.
- **التكاملية مع المؤسسة:** تصبح عملية التكاملية في معظم الأحيان مؤمنة تلقائياً في هذا البنيان كونها تعتمد الانترنت.
- **تمكين التوزيع العريض:** وذلك لكون عملية أقلمة المكونات لتناسب المستعرضات على الأجهزة المحمولة المختلفة مسؤولية المخدم.
- **البيانات الحديثة:** تؤمن هذا البنيان كون البيانات بنسختها الأخيرة كون المصدر لهذه البيانات هومخدم الوب ولن يكون هناك تخزين على طرف الزبون.
- **الأمان:** كون البيانات مخزنة على المخدم.

المساوئ :

- **الاتصال اللاسكي: تتطلب** عملية الوصول إلى البيانات الاتصال بالمخدم لاسلكياً مما يشكل أحياناً صعوبة وبالأخص أثناء التنقل بين الأماكن المختلفة.
- **واجهة المستخدم البسيطة:** غالباً ما توفرالمستعرضات الصغرية قدرات بيانية محدودة مما يجعل من غير المناسب استخدام الصور.
- **أداء التطبيق:** يرتبط أداء التطبيق بشكل كبير بأداء وفعالية الشبكة.
- **اختبار التطبيق:** تكون عملية اختبار التطبيق مشكلة في مثل هذا البنيان مع التنوع الكبير في المستعرضات الصغرية. إذ لا تكون الاختبارات دقيقة بالقدر المطلوب حتى بوجود برمجيات المحاكاة للتجهيزات.
- **الجاهزية:** تسبب أي مشكلة قد تطرأ على المخدم توقف عمل جميع المستخدمين.
- **الأمان:**لا يوجد سيطرة كاملة على البيئة وذلك بسبب وجود العبّارة اللاسلكية والتي قد تعتبر أحد نقاط الضعف.
- **الكلفة:** تصبح الكلفة في كثير من الأحيان عائقاً في مثل هذا البنيان بسبب اضطرار المستخدم إلى تأسيس اتصال لاستخدام التطبيقات.

من أهم الأمثلة على التطبيقات المناسبة لهذا البنيان هي:

1- تجارة السندات والبورصة: حيث تبرز أهمية الحصول على أحدث البيانات حول الأسعار الآنية كون عملية التحديث تتم آنياً على مخدم الوب.
2- خدمات المعلومات.
3- خدمات التسلية : كالأبراج والألعاب والأحاجي.
4- التجارة المحمولة.

بنيان الزبون الذكي

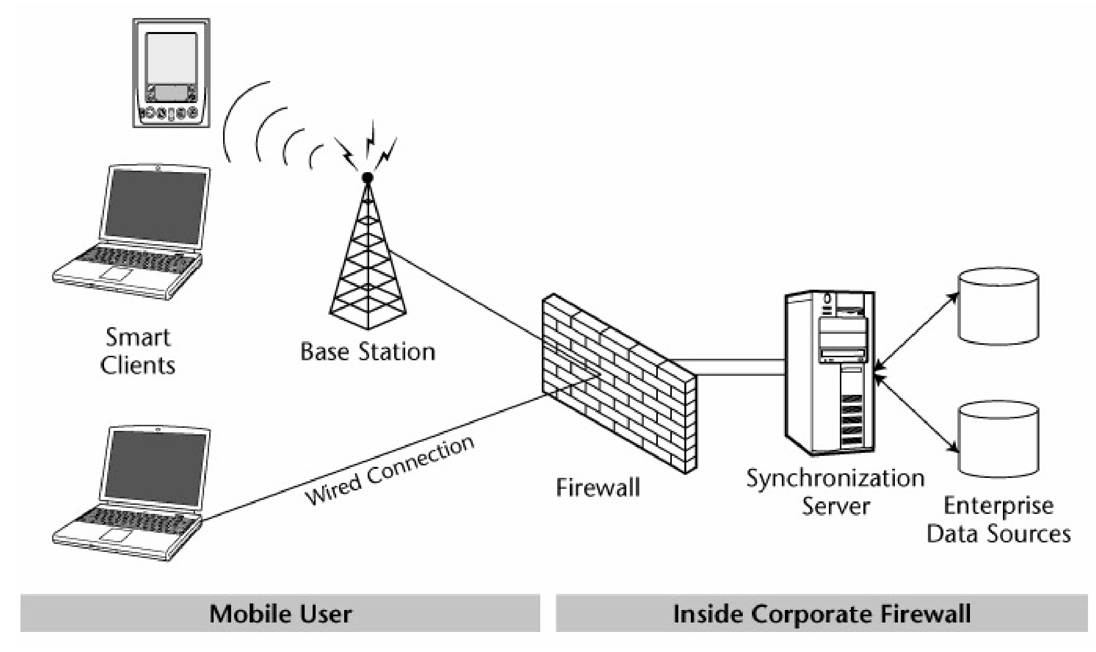
يشكل بنيان الزبون الذكي بديلاً قوياً لبنيان تطبيقات الانترنت اللاسلكي.

تعتمد هذه التقنية على تطوير برنامج خاص يتم تثبيته في طرف الزبون يحتوي هذا البرنامج عادةً على قاعدة البيانات إضافةً إلى منطق العمل.

يعني وجود هذا البرنامج على طرف الزبون إمكانية عمل الزبون في أي مكان أوزمان دون الحاجة إلى الاتصال.

تتم عملية التكامل مع المؤسسة بإجراء عملية مزامنة حيث يتصل التطبيق الزبون بقاعدة البيانات في المؤسسة عبر مخدم مزامنة.

يمكن لعملية الاتصل بغرض المزامنة أن تتم عن طريق اتصال tcp/ip، اتصال تسلسلي أو USB.



يحتاج تثبيت برنامج على الجهاز المحمول إلى توفر نظام تشغيل لدعم التطبيق مع ضرورة وجود مكتبات خاصة تابعة لنظام التشغيل هذا ليقوم التطبيق باستخدامها، كما يمكن أن يكون استخدام تطبيقات مطورة بلغة جافا بديلاً مناسباً أيضاً.

مكونات بنيان الزبون الذكي

الزبون الذكي: تطبيق الزبون الذي يحتوي على منطق العمل.
مخدم المزامنة: يتم إرسال البيانات من التطبيق الزبون إلى مخدم المزامنة الذي تتلخص مهمته في مزامنة بيانات المؤسسة مع بيانات التطبيق الزبون حيث يقوم بتأمين هذه العملية بالحد الأدنى من البيانات المنقولة.
مصدر البيانات الخاص بالمؤسسة: يقوم مخدم المزامنة بالوصول إلى مصدر البيانات في المؤسسة باستخدام تقنية الاتصال المفضلة.
يمكن أن تستلزم عملية المزامنة الاتصال بمخدم المزامنة لحين انتهاء المؤسسة من معالجة البيانات.

محاسن ومساوئ بنيان الزبون الذكي:

محاسن بنيان الزبون الذكي:

1- **التوفر الدائم للبيانات:** حتى في حال عدم وجود اتصال لاسلكي يستمر المستخدم بالعمل وبتعديل معلومات المؤسسة على أن تتم مزامنة هذه التغيرات لاحقاً.
2- **غنى واجهة الاستخدام:** حيث يمكن تطوير واجهات رسومية وبيانية للتطبيق الذكي في حال حواسب الجيب الشخصية مثلاً. كما يمكن اعتماد مجموعة جزئية من الوظائف التي يقوم بها نظام التشغيل WINDOWS لتوفير مكونات بيانية غنية.
3- **الأداء:** لن تتم إضاعة الأداء في محاولة اتصال مستمرة للوصول إلى البيانات بل يعتمد الأداء على سرعة المعالج وسرعة الوصول إلى البيانات على الجهاز المحمول.
4- **المعالجة الموزعة:** إن عملية تنفيذ منطق العمل من طرف المخدم تخفف من عبء العمل عن المخدم مما يسمح للمؤسسة بشراء مخدمات أقل كلفة.
5- **الأمان:** يمنح تطبيق آلية أمان بين النهايات (المخدم والزبون) تحكم أكبر وسيطرة على طرفي تراسل البيانات ولا يكون هناك تخوف من نقاط الضعف التي فرضتها العبّارة في بنيان الانترنت اللاسلكية.
6- **الكلفة:** يجري في هذا البنيان تخفيف كلف الاتصال اللاسلكي للحد الأدنى وذلك بسبب الحاجة للاتصال فقط عند إجراء عملية المزامنة ولمدة قصيرة بحيث يتم تقليل كمية البيانات المتبادلة أثناء المُزامنة للحد الأدنى.

مساوئ بنيان الزبون الذكي:
المكاملة مع المؤسسة: تتطلب عملية المكاملة بين تطبيقات الزبون الذكي وتطبيقات المؤسسة وقتاً أكثر والمزيد من الموارد نسبة إلى تلك اللازمة في حالة بنيان الانترنت اللاسلكي.

تثبيت البرنامج من طرف الزبون: تُعتبر عملية تثبيت وإدارة التطبيق على الجهاز الزبون عملية غير صعبة، ويمكن إجراؤها عن بعد ولكنها تصبح عبئاً في حال وجود عدد كبير من الزبائن.

فيروسات الموبايل: يفتح استخدام نظام تشغيل خاص بالتجهيزات المحمولة المجال أمام خطر الفيروسات. بالرغم من وجود حلول لهذه المشكلة بحيث يمكن تطبيقها أثناء تثبيت التطبيق الزبون إلا أن وجود المشكلة بحد ذاته أضاف مزيداً من العبء.
تعقيد عملية التطوير: تجري عملية تطوير التجهيزات المحمولة إنجازها باستخدام واجهات تطبيقات برمجية تابعة لنظام التشغيل

الخاص بجهاز محمول بعينه، مما يجعل عملية تطوير برمجيات تابعة لمجموعة كبيرة من التجهيزات مشكلة حقيقية. وفرت اللغات المستقلة عن منصة العمل مثل Java حلاً لا بأس به لهذه المشكلة.

تعدد دورات التطوير: في حال تعدد أنواع التجهيزات وأنظمة التشغيل تبرز أهمية إنشاء دورات تطوير واختبار مختلفة لكل نوع من هذه التجهيزات مما يستنزف الموارد.

تطبيقات بنيان الزبون الذكي

من أهم تطبيقات بنيان الزبون الذكي التطبيقات التالية:

1- فرق المبيعات المؤتمتة:

تعد عملية الحصول على المعلومات الحالية من أهم عناصر نجاح عملية البيع. إذ تمثل إمكانية وصول مندوب المبيعات إلى المعلومات أثناء مقابلته للزبون، نقطة الفصل أحياناً في إتمام هذه العملية.

تُمكِّن تطبيقات فرق المبيعات المؤتمتة أوما يسمى (SFA)، مندوب المبيعات من الوصول إلى معلومات مختلفة كمعلومات خاصة بالزبون، ومعلومات عمليات الشراء التي قام بها مسبقاً، ومعلومات متعلقة بكاتالوجات المنتجات، ولوائح الأسعار، ولوائح الموجودات في المستودع، وكذلك بعمليات البيع الحاصلة، حيث يمكن للمندوب مزامنة المعلومات كلما استدعت الحاجة.

2- تطبيقات فرق العمل الحقلية وعمليات جمع البيانات:

تساعد تطبيقات فرق العمل الحقلية على تمكين جمع البيانات ومزامنتها مع بيانات المؤسسة دون الحاجة إلى الدخول في الأعمال الورقية التي تستهلك الوقت.

3- تطبيقات العناية الصحية:

توفر مثل هذه التطبيقات معلومات تفصيلية عن المرضى للأطباء واتسهل أعمال كتابة الوصفات ومعالجتها مع شركات التأمين الصحي.

4- تطبيقات الانتاجية المحمولة :

تساعد هذه التطبيقات الأشخاص العاملين في مجال الخدمات على تسجيل ساعات العمل التي قضوها في كل مهمة من المهام ومكاملة هذه السجلات لدى المؤسسة.

بنيان التراسل

يمكن للتطبيقات التي تعتمد بنيان التراسل أن تأخذ العديد من الأشكال، التي تتدرج من شكل البريد الالكتروني والتنبيه إلى سوية التراسل تطبيق إلى تطبيق.

تستخدم عملية التراسل عادة إما كميزة إضافية للتطبيقات المتوفرة أو كبنيان مستقل للتطبيق.

تتوفر العديد من تقنيات التراسل وفيما يلي نقدم لأهمها:

1- التراسل من نمط مستخدم لمستخدم: يتم في هذه التقنية إرسال الرسائل من مستخدم إلى آخر باستخدام العديد من الآليات أهمها البريد الالكتروني، أوالتنبيه، أوالرسائل النصية اللاسلكية مثل رسائل SMS، أوالتراسل المباشر IM، في حين يمكن إرسال الرسائل التي تحتوي مكونات أغنى كالصور والنص المنسق باستخدام خدمة الرسائل المُحسَّنة EMS، أما بالنسبة للرسائل الحاوية على مكونات متعددة الوسائط فيمكن استخدام خدمة الرسائل متعددة الوسائط MMS.

يمكن أن يتم توليد هذه الرسائل عن طريق إجرائيات من طرف المخدم كطريقة لتوزيع الرسائل.

2- التنبيهات ورسائل الإخطار:

يشمل هذا النوع الرسائل ذات الطبيعة العاجلة بطبيعتها والتي يمكن إرسالها إلى المستخدمين على أجهزتهم المحمولة. تُمكِّن هذه التقنية الشركات من ضمان وصول المعلومات بشكل آني.

من أهم التقنيات المستخدمة في هذا المجال تبيهات HDML و WAP Push.

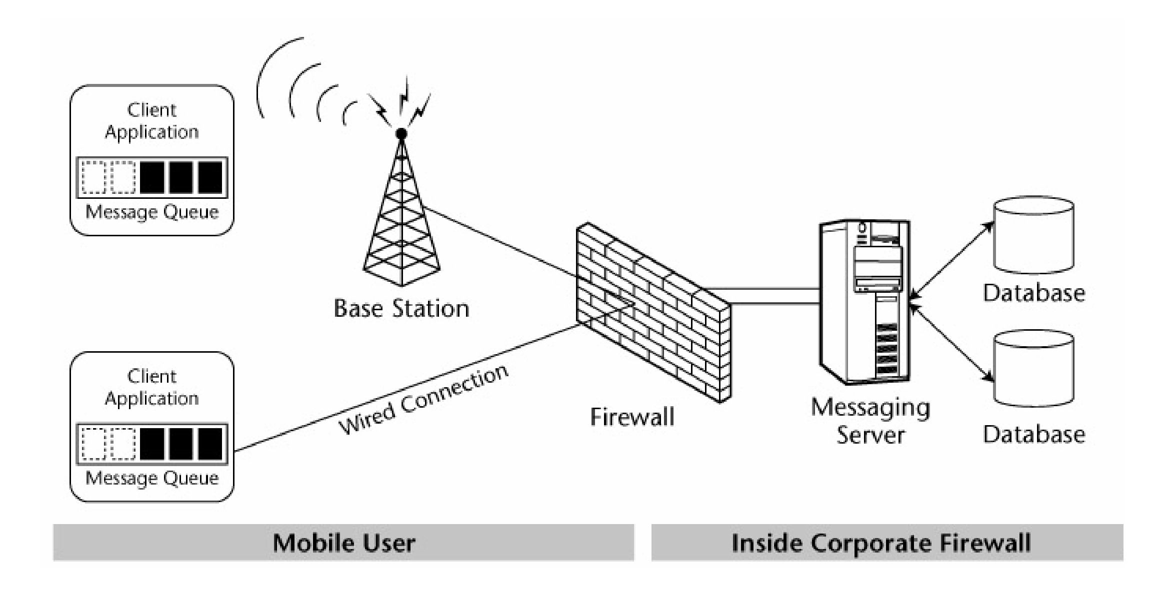
3- التراسل من نمط تطبيق إلى تطبيق

في الكثير من الحالات لا يكون تفاعل المستخدم مطلوباً لنجاح التراسل إذ يكفي أن يجري الاتصال بين مخدم المؤسسة والتطبيق الزبون دون الحاجة إلى تدخل المستخدم.

يقع هذا النوع من التراسل ضمن بنيان الزبون الذكي لأنه يحتاج أيضاً إلى تثبيت تطبيق من جهة الزبون.

يمكن للمستخدم أن يقوم بإرسال رسالة باستخدام التطبيق المثبت على الجهاز دون الحاجة إلى الاتصال في ذلك الوقت. يمتلك التطبيق الزبون رتل انتظار تتم فيه مراكمة المعلومات.

فيما يلي شكل توضيحي يبين تقنية التراسل من تطبيق إلى تطبيق.



بنيان التراسل

مكونات بنيان التراسل من تطبيق إلى تطبيق:

زبون التراسل:
يعبر عن التطبيق الحاوي على رتل الرسائل، إضافةً إلى منطق العمل من جهة الزبون.

مخدم التراسل:
يمثل مكون المخدم جزء النظام الذي يقوم بالاتصال بزبون التراسل من جهة، وبمصدر البيانات في المؤسسة من طرف آخر.
يطلق على مخدم التراسل أحياناً اسم الإطار الوسيط الخاص بالرسائل MOM.
يتم تطوير معظم هذه الأنظمة باستخدام خدمة جافا للرسائل JMS حيث تزود تلك الأخيرة منصة عمل ذات اعتمادية ومرونة عالية لتطوير مثل هذه الأنظمة.

مصدر البيانات في المؤسسة:
يمكن لمخدم التراسل الاتصال والتفاعل مع مجموعة كبيرة من مصادر البيانات كقواعد البيانات وتطبيقات الأعمال وأنظمة التراسل الأخرى.

محاسن ومساوئ بنيان التراسل

المحاسن:
إمكانيات الدفع: يمكن تحسين عمل تطبيقات الأنترنت اللاسلكي والزبون الذكي من خلال آلية دفع الرسائل. فمثلاً بالنسبة للانترنت اللاسلكي، يمكن إخطار المستخدم بضرورة إجراء عملية المزامنة في حال بنيان الزبون الذكي.

التخزين والتوجيه: يتم في هذا البنيان تخزين الرسائل ووضعها في رتل انتظار أي يمكن أن يعمل المستخدم على إرسال الرسالة حتى في حالة عدم الاتصال حيث تتم إعادة توجيه الرسائل عند تأسيس الاتصال.

الاتصال السلكي واللاسكي: يمكن لهذه التقنية أن تعمل باستخدام الاتصال السلكي أواللاسلكي في حين يتطلب التراسل من نمط مستخدم إلى مستخدم الاتصال اللاسلكي.

المساوئ:
تقنية التراسل هي تقنية كيفية غير إجبارية وعادة لا تتم إضافتها ما لم تثبت فائدة واضحة لذلك لا تتضمن الحلول التي تعتمد هذه التقنية مساوئ واضحة فيما خلا زيادة التطبيقات تعقيداً.

على أي حال تعتبر هذه التقنية شعبية بين المستخدمين ويتم اعتمادها في التطبيقات المختلفة سواء تلك المخصصة للاتصال بين الأشخاص أولأغراض التسلية كإرسال نتائج المباريات وتنبؤات الأبراج كذلك في الأغراض المرتبطة بالأحداث الاقتصادية وأسعار البورصة والأسهم.

القسم الخامس والسادس:

لغات التأشير وتقنيات توليد المحتوى

الكلمات المفتاحية:

تطوير، المحتوى، لغة تأشير، تأشيرة، واصفة، تعبير.

ملخص:

سنحاول في هذه الجلسة تغطية موضوعين أساسيين الأول يتناول لغات التأشير المستخدمة في تطوير التطبيقات اللاسلكية والثاني يتناول تقنيات توليد المحتوى.

أهداف تعليمية:

يتعرف الطالب في هذا الفصل على:

- لغات التأشير المُستخدمة في تطوير التطبيقات اللاسلكية
- تقنيات توليد المحتوى.

اللغات اللاسلكية وتقنيات توليد المحتوى

نغطي في هذه الجلسة موضوعين أساسيين: يتناول الأول لغات التأشير المستخدمة في تطوير التطبيقات اللاسلكية، ويتناول الثاني تقنيات توليد المحتوى.

أنماط المحتوى اللاسلكي

يدخل في إنشاء تطبيق لاسلكي مجموعة من الأنواع المختلفة للمحتوى بما فيها لغات التأشير، والصور، والوسائط المتعددة، والملفات المختلفة المتعلقة بعمل التطبيق.

غالباً، لا نحتاج –في معظم التطبيقات الخاصة بالهواتف المحمولة أوالتجهيزات التي تدعم WAP– إلا إلى التركيز على لغة التأشير وبعض الصور البسيطة، أما إذا كان السوق المستهدف بالتطبيق يتضمن التجهيزات القادرة على تشغيل أنظمة كـــ Windows CE أوSymbian OS فلا بد عندها من التركيز على عناصر إضافية كعناصر الوسائط المتعددة، وملفات التطبيقات المختلفة كـــMSWord.

لغات التأشير:

عندما نقوم بتطوير تطبيقات إنترنت مكتبية نلجأ غالباً إلى لغة تأشير.

من بين لغات التأشير الأكثر شيوعاً في هذا المجال لغة HTML فهي قادرة على تمثيل المحتوى لأغلب المستعرضات مثل IE، Mozilla، Netscape...

للأسف هذه ليست الحال بالنسبة للعالم اللاسلكي حيث يمكن يتم استخدام لغات مثل HDML، WML، HTML، CHTML وXHTML على مستعرضات مختلفة، والعوامل الذي تحدد اختيار أي من تلك اللغات تتعلق بالموقع الجغرافي، ونوع الجهاز، والمستعرض الصغري المستخدم.

لغة HDML

يتركز استعمال لغة HDML بصورة أساسية في منطقة أمريكا الشمالية نظراً لكون أغلب التجهيزات المحمولة المطورة للاستخدام في أمريكا الشمالية تدعم هذه اللغة. وقد كانت الفكرة خلف تطوير هذه اللغة، إنشاء لغة تأشير خاصة بالهواتف المحمولة.

تتشابه HDML مع HTML من حيث البنية لكنها لا تقترب من الإمكانيات الواسعة التي تمنحها HTML.

تكمن أحد أهم نقاط ضعف HDML في أنها لا توفر لغة خطاطية كالتي توفرها HTML مثل Javascript أوVBscript، أوكالتي توفرها WML مثل WMLScript، مما يجعل HDML غير قادرة إلا على تنفيذ المنطق المتوفر على الجهاز المحمول المُستَخدَم ليس إلا.

لذلك فإن عمليات مثل التحقق من الإدخال، وتوليد الرسائل، وتوليد مربعات الحوار، لا يمكن تطبيقها إلا كنصوص برمجية من جهة المخدم ولا يمكن تنفيذها من جهة الزبون مما يسبب بالطبع ازدياد كميات البيانات الواجب نقلها على الشبكة اللاسلكية وبالتالي إلى خفض الأداء.

لكن، بالرغم من المآخذ على هذه اللغة، ليس بالإمكان التغاضي عن استخدامها وبالأخص عندما نعلم أن ملايين التجهيزات في أمريكا

وكندا تستخدم المستعرضات العاملة على HDML.

يتلخص أحد الحلول المطروحة في إضافة عبّارة تقوم بتحويل HDML إلى WML بحيث يصبح بالإمكان التعامل مع التطبيقات الخاصة بـــ HDML باستخدام التجهيزات التي تدعم WML.

مثال عن لغة HDML:

فيما يلي مثال عن نص برمجي بلغة HDML مع النتيجة التي يظهرها.



النص البرمجي:

```
1. <HDML VERSION="3.0">
2.   <CHOICE>
3.     <CENTER><b>Inventory Search</b>
4.     <CE TASK="GO" DEST="#ProductSearch">Search by Name
5.     <CE TASK="GO" DEST="#SKUSearch">Search by SKU
6.     <CE TASK="GOSUB" DEST=inventorylist.jsp>Inventory List
7.   </CHOICE>
8.   <ENTRY NAME="ProductSearch" KEY="ProductName">
9.     <ACTION TYPE="ACCEPT" TASK="GO"
      DEST="ProductSearch.jsp?Product=$ProductName">
10.    Enter Product Name:
11.  </ENTRY>
12.  <ENTRY NAME="SKUSearch" KEY="SKU">
13.    <ACTION TYPE="ACCEPT" TASK="GO" DEST="SKUSearch.jsp?SKU=$SKU">
14.    Enter SKU:
15.  </ENTRY>
16.</HDML>
```

نلاحظ أن <HDML> تبدأ برقم النسخة.

يُقسم النص البرمجي في hdml إلى ما يسمى منصات وبطاقات.

يعبّر المثال السابق عن محتوى منصة وهي القطعة من النص البرمجي التي يجري تحميلها.

- تحتوي كل منصة بطاقة أومجموعة بطاقات، ففي حالة مثالنا لدينا ثلاث بطاقات هي تلك المحددة بالتأشيرات <Choice> و<Entry>
- يتم التعامل مع كل بطاقة على أنها صفحة منفصلة
- تسمح HDML بالانتقال بين البطاقات المختلفة
- نلاحظ في مثالنا أنه تم استخدام التأشيرة <CHOICE> للحصول على دخل من المستخدم، وهي الطريقة الأفضل للإدخال نظراً لصعوبة عملية الإدخال على التجهيزات المحمولة كما ذكرنا
- تم استخدام الواصفة DEST لتحديد البطاقة أوالمنصة الهدف.
- تم استخدام التأشيرة <Entry> للحصول على دَخل من المستخدم عن طريق لوحة المفاتيح.

لمزيد من المعلومات حول هذه اللغة يمكن العودة إلى الوصلة التالية
http://demo.openwave.com/pdf/styleguides/hdml_style.pdf.

يتركز استعمال لغة HDML بصورة أساسية في منطقة أمريكا الشمالية نظراً لكون أغلب التجهيزات المحمولة المطورة للاستخدام في أمريكا الشمالية تدعم هذه اللغة. وقد كانت الفكرة خلف تطوير هذه اللغة، إنشاء لغة تأشير خاصة بالهواتف المحمولة.

تتشابه HDML مع HTML من حيث البنية لكنها لا تقترب من الإمكانيات الواسعة التي تمنحها HTML.

تكمن أحد أهم نقاط ضعف HDML في أنها لا توفر لغة خطاطية كالتي توفرها HTML مثل Javascript أوVBscript، أوكالتي توفرها WML مثل WMLScript، مما يجعل HDML غير قادرة إلا على تنفيذ المنطق المتوفر على الجهاز المحمول المُستَخدَم ليس إلا.
لذلك فإن عمليات مثل التحقق من الإدخال، وتوليد الرسائل، وتوليد مربعات الحوار، لا يمكن تطبيقها إلا كنصوص برمجية من جهة المخدم ولا يمكن تنفيذها من جهة الزبون مما يسبب بالطبع ازدياد كميات البيانات الواجب نقلها على الشبكة اللاسلكية وبالتالي إلى خفض الأداء.

لكن، بالرغم من المآخذ على هذه اللغة، ليس بالإمكان التغاضي عن استخدامها وبالأخص عندما نعلم أن ملايين التجهيزات في أمريكا وكندا تستخدم المستعرضات العاملة على HDML.

يتلخص أحد الحلول المطروحة في إضافة عبّارة تقوم بتحويل HDML إلى WML بحيث يصبح بالإمكان التعامل مع التطبيقات الخاصة بـــ HDML باستخدام التجهيزات التي تدعم WML.

لغة WML

سبق وذكرنا أن لغة WML هي جزء من بيئة التطبيقات اللاسلكية WAE حسب ما تم تعريفه في البروتوكول WAP.

تم تأسيس WML –بعكس HDML– على لغة XML لذا فإن قواعد الصياغة فيها أكثر صرامة.

تُعتبر WML خليفة HDML، وقد تم تطويرها بصورة أساسية لتكون موجهة إلى التجهيزات المحمولة كأجهزة PDA، والهواتف الخليوية، والهواتف الذكية.

تلقى WML الدعم الكامل من معظم شركات التجهيزات المحمولة مما يجعلها الخيار الأفضل كلغة تأشير في تطوير التطبيقات المحمولة. ولكن مع ظهور WAP2.0 والتي تعتمد XHMTL كلغة تأشير لا نعلم كما ستستمر هذه اللغة في الاستخدام.

مثال عن لغة WML:

سنستعرض فيما يلي مثالاً يقدم نفس الوظيفة التي رأيناها في المثال الخاص بـHDML ولكن باستخدام لغة WML هذه المرة.



النص البرمجي:

```
1.  <?xml version="1.0" encoding="UTF-8"?>
2.  <!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.1//EN"
"http://www.wapforum.org/DTD/wml_1.1.xml">
3.  <wml>
4.    <card id="card1">
5.      <p align="center"><i>Inventory Search</i></p>
6.      <p align="left">
7.      <select>
8.        <option>Search by Name
9.          <onevent type="onpick">
10.           <go href="#ProductSearch"></go>
11.         </onevent>
12.       </option>
```

```
13.        <option>Search by SKU
14.          <onevent type="onpick">
15.            <go href="#SKUSearch"></go>
16.          </onevent>
17.        </option>
18.        <option>View Inventory List
19.          <onevent type="onpick">
20.            <go href="Inventorylist.wml"></go>
21.          </onevent>
22.        </option>
23.      </select>
24.      </p>
25.    </card>
26.    <card id="ProductSearch">
27.    <!--WML code here for Product Search-->
28.      <p>
29.        Enter Product Name:
30.        <input name="product" emptyok="false"></input>
31.      </p>
32.    </card>
33.    <card id="SKUSearch">
34.    <!--WML code here for SKU Search-->
35.      <p>
36.        Enter SKU:
37.        <input name="sku" emptyok="false"></input>
38.      </p>
39.    </card>
40.  </wml>
```

تحافظ WML على صيغة المنصات، والبطاقات المستخدمة في HDML، ومثالنا كما نلاحظ يحتوي أيضاً منصة واحدة وثلاث بطاقات.

يمكننا ملاحظة الصيغة التي تشبه صيغة ملف XML:

- تم التصريح كما نرى أولاً عن نوع الوثيقة
- تم التصريح عن البطاقة بالتأشيرة <card> وتم استخدام التأشيرة <select> للحصول على خيار المستخدم
- تم استخدام التأشيرة <input> للحصول على دخل المستخدم بشكل مشابه لاستخدام التأشيرة <Entry> في مثال HDML
- هناك تشابه كبير بين التأشيرات في WML وتأشيرات HTML
- تقدم WML عدة طرق للانتقال بين البطاقات والمنصات منها الطريقة التي يقدمها المثال باستخدام التأشيرة <go> مع التأشيرة <onevent>

أبرز ما يجب التركيز عليه في WML هوالصرامة الشديدة في التعامل مع التأشيرات إذ لا بد من إغلاق كل التأشيرات، ويجب تلافي وجود أي تقاطعات بين التأشيرات (يجب أن تكون مُعلّبة تماماً ببعضها البعض) كما يجب أن تستخدم التأشيرات الحروف الصغيرة.

WMLScript

تتضمن WML لغة خطاطية تسمى WMLScript وهي تقابل javascript في HTML.

تمكِّن WMLScript من إضافة منطق عمل من طرف الزبون، مما يقدم الكثير من الميزات كتقليل زمن الانتظار، وتخفيف العبء على الشبكة اللاسلكية.

من أهم استخدامات WMLScript:

- عمليات التحقق من الإدخال
- إمكانية الوصول إلى التسهيلات التي يقدمها الجهاز: كإمكانية إضافة اسم، أورقم إلى دليل الأرقام على الجهاز
- رفع سوية التفاعل مع المستخدم بتوليد الرسائل ومربعات الحوار من طرف الزبون دون الحاجة إلى المخدم في هذا العمل

لكي نتمكن من استخدام لغة WMLScript في التطبيقات، يجب أن توضع النصوص البرمجية التي نود استخدامها في ملفات تملك اللاحقة wmls.

مثال:

فيما يلي مثال بسيط عن استخدام WMLScript

```
<a href="validateuser.wmls#foo($(user))>Validate User</a>
```

نرى هنا أننا استخدمنا التابع foo من الملف validateuser.wmls.

للمزيد من المعلومات حول هذه اللغة يمكن العودة إلى الوصلة التالية
http://demo.openwave.com/pdf/50/wmls_dev_guide.pdf

تتضمن WML لغة خطاطية تسمى WMLScript وهي تقابل javascript في HTML.

تمكِّن WMLScript من إضافة منطق عمل من طرف الزبون، مما يقدم الكثير من الميزات كتقليل زمن الانتظار، وتخفيف العبء على الشبكة اللاسلكية.

من أهم استخدامات WMLScript:

- عمليات التحقق من الإدخال
- إمكانية الوصول إلى التسهيلات التي يقدمها الجهاز: كإمكانية إضافة اسم، أورقم إلى دليل الأرقام على الجهاز
- رفع سوية التفاعل مع المستخدم بتوليد الرسائل ومربعات الحوار من طرف الزبون دون الحاجة إلى المخدم في هذا العمل

لكي نتمكن من استخدام لغة WMLScript في التطبيقات، يجب أن توضع النصوص البرمجية التي نود استخدامها في ملفات تملك اللاحقة wmls

لغة HTML

يمتلك معظم المطورون خبرة ممتازة في استخدام لغة HTML لذلك لن ندخل في تفاصيلها، ولكن يكفي القول هنا أن HTML تُعتبر من لغات التأشير الأساسية التي تقدم مرونة كبيرة تمنح المطور إمكانيات متعددة لتطوير التطبيقات التي نحن بصددها.

تتعامل HTML كما نعلم مع لغات خطاطية هي VBScript و JavaScript وهي لغات تمكن الزبون من القيام بجزء من العمل دون الحاجة إلى إعادة تحديث المحتوى من المخدم.

مثال:

فيما يلي مثال مشابه لما قمنا به باستخدام اللغات HDML و WML ولكن باستخدام لغة التأشير HTML:

```
1.  <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2 Final//EN">
2.  <html>
3.  <head>
4.    <title>Inventory List</title>
5.  </head>
6.  <body>
7.  <P align=left><FONT size=4><STRONG>Inventory Product
List</STRONG></FONT>
8.  <br>
9.  <br>
10.  <TABLE cellPadding=3 border=1>
11.    <TR>
12.      <TD><STRONG>Product </STRONG></TD>
13.      <TD><STRONG>Quantity</STRONG></TD>
14.      <TD><STRONG>Price($)</STRONG></TD>
15.    </TR>
16.    <TR>
17.      <TD>Sony TRV30
18.      <TD>17
19.      <TD>1699.99
20.    </TR>
21.    <TR>
22.      <TD>Hitachi VMD875L
23.      <TD>24
24.      <TD>599.99
25.    </TR>
26.    <TR>
27.      <TD>Sony DCR-IP7BT
28.      <TD>11
29.      <TD>2199.99
30.    </TR>
31.    <TR>
32.      <TD>JVC GR-DV2000
```

```
33.         <TD>4
34.         <TD>1599.99
35.       </TR>
36.    </TABLE>
37. </BODY>
38. </HTML>
```

تظهر نتيجة هذا النص البرمجي على مستعرض Microsoft Mobile Internet Explorer كما في الشكل التالي:



يمتلك معظم المطورون خبرة ممتازة في استخدام لغة HTML لذلك لن ندخل في تفاصيلها، ولكن يكفي القول هنا أن HTML تُعتبر من لغات التأشير الأساسية التي تقدم مرونة كبيرة تمنح المطور إمكانيات متعددة لتطوير التطبيقات التي نحن بصددها.

تتعامل HTML كما نعلم مع لغات خطاطية هي VBScript و JavaScript وهي لغات تمكن الزبون من القيام بجزء من العمل دون الحاجة إلى إعادة تحديث المحتوى من المخدم.

لغة CHTML

تعد CHTML من أكثر لغات HTML الجزئية شعبية، وتسمى HTML المُدمجة ويعود سبب نجاحها لاستخدامها في خدمة i-Mode المُقدَّمة من أكبر شركات الاتصالات في اليابان NTT DOCOMO حيث تساعد هذه الخدمة المستخدمين في الوصول إلى محتوى الانترنت من خلال أجهزتهم المحمولة.

لم يجر إضافة أي جديد إلى هذه اللغة وفي الحقيقة كانت عبارة عن لغة انتقالية تمهيداً للانتقال بين HDML و WML إلى XHTML.

عند إنشاء هذه اللغة تم أخذ العوامل التالية بعين الاعتبار:

- محدودية التجهيزات بما فيها محدودية الذاكرة، والتوجه إلى خفض استهلاك الطاقة في عمليات المعالجة من قبل الــCPU، إضافةً إلى مساحة الإظهار المحدودة
- محدودية عملية الحركة بين الخيارات، والمعلومات، ومحاولة إجراء العمليات بأقل عدد من الخطوات
- الاستقلالية عن الشبكة اللاسلكية المستخدمة

تم تصميم CHTML بشكل كامل اعتماداً على توصيات W3C وكان التركيز بشكل أساسي على جعلها خفيفة تستطيع التعامل حتى مع إمكانيات الإظهار المحدودة، وعلى جعلها سهلة التطبيق.

أهم العناصر التي دعمتها هذه اللغة:

- صور JPEG
- الأنواع والأنماط المختلفة للخطوط
- الخلفيات (كألوان أوصور)
- صفحات الأنماط CSS

بالرغم من اعتماد CHTML على HTML إلا أنها فقدت الكثير من أهميتها نتيجة لعدم إنشاؤها على أساس XML.

لمزيد من المعلومات حول هذه اللغة يمكن العودة إلى الرابط التالي
.www.w3.org/TR/1998/NOTE-compactHTML-19980209

لغة XHTML

تُعتبر لغة XHTML نسخة من HTML وتعتمد XML، لذلك تتجه معظم التقنيات التي تعتمد لغات تأشير –ومنها الانترنت اللاسلكي للتجهيزات المحمولة– إلى محاولة دعم هذه اللغة لما تقدمه من إمكانيات موسعة.

من أهم المجالات التي تمنح فيها هذه اللغة ميزات جيدة، مجال تطوير تطبيقات الأعمال الالكترونية المحمولة والأعمال المحمولة.

تحولت نسخة XHTML1.0 إلى أحد المعايير الرسمية لــ W3C في عام 2000، وهي مطابقة ل HTML4.01 ولكن مع إضافة بعض القواعد لضمان توافقها مع بنية XML.

لماذا XHML؟

جاءت XHTML بشكل أساسي لتقيّد بصورة أكبر لغة HTML بحيث تتمكن حتى المستعرضات التي لا تملك إمكانيات اكتشاف وتجاوز الأخطاء من العمل معها دون مشكلة.

كيف تختلف XHTML عن HTML؟

تختلف XHTML عن HTML في النقاط الأساسية التالية:

- يحب أن يجر تنسيق صفحات XHTML بشكل جيد وفقاً لقواعد XML
- يجب أن يجر تعليب التأشيرات ببعضها البعض بشكل صحيح دون تقاطع
- يجب أن تستخدم التأشيرات والواصفات الحروف الصغيرة
- يجب إغلاق جميع التأشيرات في XHTML حتى تلك التي لا تستخدم تأشيرة إغلاق في HTML
 التي تُكتب في XHTML
- يجب أن يجر حصر جميع قيم التأشيرات ضمن إشارتي < >
- يجب التصريح عن نوع الوثيقة.

مثال:

نعيد فيما يلي نفس المثال الوارد في الشرائح السابقة ولكن بلغة XHTML

```
1.  <?xml version="1.0"?>
2.  <!DOCTYPE html PUBLIC "-//OPENWAVE//DTD XHTML Mobile 1.0//EN"
"http://www.openwave.com/dtd/xhtml-mobile10.dtd">
3.  <html xmlns="http://www.w3.org/1999/xhtml"xml:lang="en">
4.  <head>
5.    <title>Inventory List</title>
6.  </head>
7.  <body>
8.  <p align="left"><font size="4"><strong>Inventory Product
List</strong></font></p>
9.  <br/>
10. <br/>
11. <table cellPadding="3" border="1">
12.   <tr>
13.     <th>Product</th>
14.     <th>Quantity</th>
15.     <th>Price($)</th>
16.   </tr>
17.   <tr>
18.     <td>Sony TRV30</td>
19.     <td>17</td>
20.     <td>1699.99</td>
21.   </tr>
22.   <tr>
23.     <td>Hitachi VMD875L</td>
24.     <td>24</td>
25.     <td>599.99</td>
26.   </tr>
27.   <tr>
28.     <td>Sony DCR-IP7BT</td>
29.     <td>11</td>
30.     <td>2199.99</td>
31.   </tr>
32.   <tr>
```

```
33.       <td>JVC GR-DV2000</td>
34.       <td>4</td>
35.       <td>1599.99</td>
36.     </tr>
37.  </table>
38.  </body>
39.  </html>
```

تُعتبر لغة XHTML نسخة من HTML وتعتمد XML، لذلك تتجه معظم التقنيات التي تعتمد لغات تأشير –ومنها الانترنت اللاسلكي للتجهيزات المحمولة– إلى محاولة دعم هذه اللغة لما تقدمه من إمكانيات موسعة.

من أهم المجالات التي تمنح فيها هذه اللغة ميزات جيدة، مجال تطوير تطبيقات الأعمال الالكترونية المحمولة والأعمال المحمولة.

تحولت نسخة XHTML1.0 إلى أحد المعايير الرسمية لـــ W3C في عام 2000، وهي مطابقة ل HTML4.01 ولكن مع إضافة بعض القواعد لضمان توافقها مع بنية XML.

لماذا XHML؟

جاءت XHTML بشكل أساسي لتقيّد بصورة أكبر لغة HTML بحيث تتمكن حتى المستعرضات التي لا تملك إمكانيات اكتشاف وتجاوز الأخطاء من العمل معها دون مشكلة.

كيف تختلف XHTML عن HTML؟

تختلف XHTML عن HTML في النقاط الأساسية التالية:

- يحب أن يجر تنسيق صفحات XHTML بشكل جيد وفقاً لقواعد XML
- يجب أن يجر تعليب التأشيرات ببعضها البعض بشكل صحيح دون تقاطع
- يجب أن تستخدم التأشيرات والواصفات الحروف الصغيرة
- يجب إغلاق جميع التأشيرات في XHTML حتى تلك التي لا تستخدم تأشيرة إغلاق في HTML ك
 التي تُكتب في XHTML
- يجب أن يجر حصر جميع قيم التأشيرات ضمن إشارتي < >
- يجب التصريح عن نوع الوثيقة.

لغة XHTML-MP

تعتمد بيئة التطبيقات اللاسلكية وفق ما هومحدد في معيار WAP2 ما يسمى بالتشكيل المحمول لـــ XHTML أوما يطلق عليه XHTML-MP.

يعتبر هذا التشكيل مجموعة جزئية من لغة XHTML حسب ما هووارد في التعريف الذي حدده W3C.

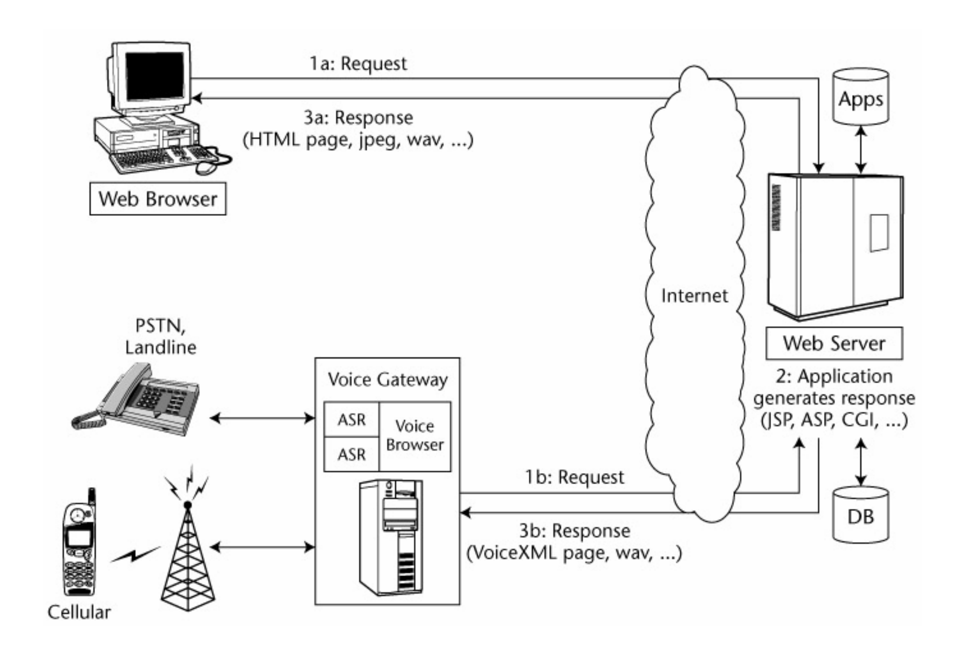
لغة VoiceXML

وهي إحدى لغات التأشير الهامة التي تستحق الذكر في هذا الجزء من موضوعنا.

هي عبارة عن لغة تأشير مبنية على XML تسمح بالوصول إلى محتوى تطبيقات الوب عن طريق الهاتف عوضاً عن مستعرض الوب. يُستخدم لهذا الغرض مستعرض صوتي يقوم بتفسير لغة VoiceXML (تماماً كما يفعل مستعرض الوب بلغات مثل XHTML أو WML)، ثم يقوم باستخدام محرك TTS الذي يقوم بتحويل النص إلى صوت ويوجه المعلومات إلى المستخدم.

تتلقى هذه اللغة الدعم من عدد كبير من الشركات العاملة في مجال الاتصالات والتقانة مثل AT&T، Motorola و IBM والعديد غيرها.

لن نخوض في التفصيلات حول هذه اللغة ولكن سنكتفي بإظهار شكل يوضح البنيان الخاص بتطبيق مثل هذه اللغة.



مثال عن VoiceXML:

سنقوم بإيراد مثال بسيط لنص برمجي بلغة VoiceXML:

```
1.<?xml version="1.0"?>
2.  <vxml version="2.0">
3.  <meta name=" author" content=" Martyn Mallick"/>
4.  <form>
5.    <block>
6.      Welcome to the voice time entry system.
7.      <goto next="#options"/>
8.    </block>
9.  </form>
10. <!-- allow user to choose one of three options -->
11. <menu id=" options" dtmf=" true">
12.   <prompt> What would you like to do? Say one of:
<enumerate/></prompt>
13.   <choice next="#entry">add entry</choice>
14.   <choice next="
```

```
http://www.timeentry.example.com/vxml/delete.vxml">
delete entry</choice>
15.   <choice next=" http://www.timeentry.example.com/vxml/list.vxml">
list entries </choice>
16.   <noinput count="1"> <reprompt/></noinput>
17.   <noinput count="2"> Please state what action you would like
<enumerate/></noinput>
18. </menu>
19. <!-- collect data for new time entry -->
20. <form id=" entry">

21.   <catch event=" nomatch noinput" count="3">
22.     <prompt> Sorry, too many attempts. Please try again later.
Goodbye.  </prompt>
23.     <throw event=" telephone.disconnect.hangup"/>
24.   </catch>
25.   <field name=" jobtype">
26.     <prompt>What is the job type for your entry? </prompt>
27.       <option>design</option>
28.       <option>development</option>
29.       <option>meeting</option>
30.       <option>travel</option>
31.       <option>vacation</option>
32.     <help>You must enter a valid job code to continue. Your options
are design, development, meeting, travel, and vacation.
<reprompt/></help>
33.   </field>
34.   <field name=" hours" type="digits">
35.     <prompt> How many hours for job <value expr="jobtype"/>?
</prompt>
36.     <help> use the keypad to enter the number of hours worked
</help>
37.   </field>
38.   <field name=" proceed" type=" boolean">
39.   <prompt>Do you want to proceed with the entry for <value
expr="hours"/> hours for job type <value expr="jobtype"/>?  </prompt>
40.     <filled>
41.       <if cond=" proceed">
42.         <prompt bargein=" false">
43.            Your entry is being entered into the time system.
44.         </prompt>
45.         <!-- submit time entry to servlet for entry into database -
-
>
46.         <submit next="/servlet/entry" namelist=" jobcode hours"/>
47.       </if>
48.       <clear namelist=" jobcode hours proceed"/>
49.       <goto next="#options"/>
50.     </filled>
51.   </field>
52. </form>
53. </vxml>
```

نلاحظ هنا أننا بدأنا بتحديد نمط الوثيقة (XML) ثم استخدمنا تأشيرة فتح للوثيقة <vxml>. لكن ما نود التركيز عليه هنا، هو أن عملية الإدخال تتم بأحد طريقتين إما بذكر الخيار بصورة صوتية، أوباختيار رقم من لوحة المفاتيح الهاتفية.

لغة XHTML-MP

تعتمد بيئة التطبيقات اللاسلكية وفق ما هومحدد في معيار WAP2 ما يسمى بالتشكيل المحمول لــ XHTML أو ما يطلق عليه XHTML-MP.

يعتبر هذا التشكيل مجموعة جزئية من لغة XHTML حسب ما هو وارد في التعريف الذي حدده W3C.

لغة VoiceXML

وهي إحدى لغات التأشير الهامة التي تستحق الذكر في هذا الجزء من موضوعنا.
هي عبارة عن لغة تأشير مبنية على XML تسمح بالوصول إلى محتوى تطبيقات الوب عن طريق الهاتف عوضاً عن مستعرض الوب. يُستخدم لهذا الغرض مستعرض صوتي يقوم بتفسير لغة VoiceXML (تماماً كما يفعل مستعرض الوب بلغات مثل XHTML أو WML)، ثم يقوم باستخدام محرك TTS الذي يقوم بتحويل النص إلى صوت ويوجه المعلومات إلى المستخدم.

تتلقى هذه اللغة الدعم من عدد كبير من الشركات العاملة في مجال الاتصالات والتقانة مثل AT&T، Motorola وIBM والعديد غيرها.

تقنيات توليد المحتوى

بعد أن اطّلعنا على أهم لغات التأشير اللاسلكية المستخدمة لا بد لنا من الاطّلاع على بعض التقنيات الخاصة بتوليد المحتوى.

تُستخدَم هذه التقنيات لإنشاء التطبيقات اللاسلكية الديناميكية. وتمتلك كل من هذه التقنيات آليات تفاعل مع بيانات المؤسسة ومع منطق العمل المتوضع على المخدم، لذلك يكون باستطاعة المطور استخدامها لإنشاء تطبيقات لاسلكية مقادة بالبيانات.

تُمكِّننا هذه التقنيات أيضاً من إنشاء استجابات مختلفة مبنية على طبيعة الجهاز المحمول الذي قام بإرسال الطلب. إذ يمكن لتطبيق واحد مثلاً الاستجابة باستخدام لغات HDML، أوWML، أوXHTML حيث تشكل هذه الميزة نقطة شديدة الأهمية وبالأخص بالنسبة للتطبيقات االاسلكية التي يجري تطويرها لجمهور واسع من المستخدمين.

مع تنوع هذه التقنيات يصبح من الصعب اختيار أحدها، ولكن غالباً ما يرتبط هذا القرار بالمهارات المتوفرة لدى المطور في إحدى هذه التقنيات، أوبمجموعة التقنيات الأخرى المستخدمة في المؤسسة خصوصاً من ناحية التطبيقات المكتبية والتي تفرض نفسها على التطبيقات اللاسلكية.

سنستعرض في هذا الجزء من الجلسة مجموعة من هذه التقنيات وهي:

- Perl – CGI

- Java servlet
- Java server page
- Active server page
- XSL Stylesheets مع XML

تقنيات توليد المحتوى
- CGI باستخدام PERL -

تعتبر تقنية CGI أحد أكثر أنواع تقنيات توليد المحتوى استخداماً وهي مدعومة تقريباً في جميع أنواع مخدمات الوب.

تحتوي CGI مجموعة من الأوامر التي تسعد في تأمين الاتصال بين مخدم الوب والبرنامج الذي يقوم بمعالجة البيانات من صفحة الوب. وتعد هذه التقنية من أولى التقنيات التي توفرت لإنشاء المواقع الديناميكية.

يجر تنفيذ ملفات Perl بالزمن الحقيقي بعكس المحتوى الثابت الذي يكون عادة بصيغة نص. ويجري، أثناء تنفيذ هذه الملفات، توليد المحتوى بشكل ديناميكي.

يمكن كتابة برامج CGI بأية لغة ولكن أكثر اللغات المستخدمة في هذا المجال انتشاراً هي ++C/C و Fortran وبالطبع Perl و Java. إذ تُعتبر لغة Perl مبنية على لغة C حيث تم تطويرها من قبل الآلاف من المطورين على مدى سنوات. وتعد هذه اللغة لغة مفسرة أي أنها لا تحتاج إلى عملية معالجة وتحويل إلى ملفات باللغة الطبيعية للآلة حيث تعد هذه ميزة نسبة إلى لغات أخرى مثل C و Fortran

تكمن أحد أهم نقاط الضعف في تقنية CGI في أن التعامل مع كل طلب يجري على حدى وتجري عملية إنشاء نسخة عن البرنامج بالنسبة لكل طلب وتخصيص مساحات للمتحولات خاصة بالنسخة مما يؤدي بالتالي استنزاف المخدم وانخفاض أدائه.

مثال:

نورد فيما يلي برنامجاً بلغة Perl يقوم بتوليد منصة WML تحتوي بطاقة وحيدة.

```
1.   #!/usr/bin/perl
2.   print "Content-type: text/vnd.wap.wml\n\n";
3.   print "<?xml version=\"1.0\" encoding=\"iso-8859-1\"?>\n";
4.   print "<!DOCTYPE wml PUBLIC \"-//WAPFORUM//DTD WML 1.1//EN\"
\" http://www.wapforum.org/DTD/wml_1.1.xml\">\n";
5.   print "<wml>\n";
6.   print "   <card id='card1'>\n";
7.   print "      <p>Hello Wireless World!</p>\n";
8.   print "   </card>\n";
9.   print "</wml>\n";
```

نلاحظ أننا لم نستخدم في هذا البرنامج سوى تعليمة Print لإرسال خرج من البرنامج حيث يجري إرسال هذا الخرج كاستجابة لطلب ما تم إرساله.

Java servlet

قامت شركة Sun Microsystem بتقديم تقنية باسم Java servlet لتجاوز عائقين: هما عائق الأداء، وعائق العمل على عدة منصات.

تجري في هذه التقنية عملية ترجمة النص البرمجي إلى رماز من نمط Bytecode والذي تجري ترجمته بدوره باستخدام آلة جافا الافتراضية JVM على مخدم الوب.

تساعد هذه التقنية في نقل النص البرمجي المكتوب بلغة Java servlet إلى أية منصة عمل تحوي JVM. أما بالنسبة للأداء فيجري إنشاء مسلك تنفيذي (Thread) لكل طلب حيث يعاد استخدام هذه المسالك عند الانتهاء منها مما يوفر الموارد.

تستخدم java servlet رماز جافا بشكل مشابه لما يحصل في CGI لتوليد المكونات ديناميكياً.

مثال:

فيما يلي مثال بسيط على استخدام تقنية Javaservlet.
سنلاحظ في هذا المثال استخدام أمر Import لاستيراد الحزمة الخاصة بـــ Servlet والتي تشكل جزء من إطار العمل الذي تم تطويره وفقاً ل J2EE. كما سنلاحظ استخدام out.println لإرسال الخرج كاستجابة للمستخدم.

```
1.  // Import the required Java libraries
2.  import java.io.*;
3.  // Import the required Java Servlet libraries
4.  import javax.servlet.*;
5.  import javax.servlet.http.*;
6.  public class HelloWorld extends HttpServlet
7.  {
8.    public void doGet(HttpServletRequest req, HttpServletResponse
res)
throws ServletException, IOException
9.    {
10.     res.setContentType("text/html");
11.     PrintWriter out = res.getWriter();
12.     out.println("<!DOCTYPE html PUBLIC \"-//OPENWAVE//DTD XHTML
Mobile 1.0//EN\" \" http://www.openwave.com/dtd/xhtml-
mobile10.dtd\">");
13.     out.println("<html
xmlns=\"http://www.w3.org/1999/xhtml\"xml:lang=\"en\">");
14.     out.println("<head>");
```

```
15.     out.println("<title>XHTML Servlet</title>");
16.     out.println("</head>");
17.     out.println("<body>");
18.     out.println("<p align=\"left\"><b>Hello XHTML Wireless
World!</b></p>");
19.     out.println("</body>");
20.     out.println("</html>");
21.   }
22. }
```

قامت شركة Sun Microsystem بتقديم تقنية باسم Java servlet لتجاوز عائقين: هما عائق الأداء، وعائق العمل على عدة منصات.

تجري في هذه التقنية عملية ترجمة النص البرمجي إلى رماز من نمط Bytecode والذي تجري ترجمته بدوره باستخدام آلة جافا الافتراضية JVM على مخدم الوب.

تساعد هذه التقنية في نقل النص البرمجي المكتوب بلغة Java servlet إلى أية منصة عمل تحوي JVM. أما بالنسبة للأداء فيجري إنشاء مسلك تنفيذي (Thread) لكل طلب حيث يعاد استخدام هذه المسالك عند الانتهاء منها مما يوفر الموارد.

تستخدم java servlet رماز جافا بشكل مشابه لما يحصل في CGI لتوليد المكونات ديناميكياً.

صفحات جافا الخاصة بالمخدم
JSP

تُعتبر JSP الحل السحري للمطور الذي يريد الحصول على ميزة الاستقلال عن منصة العمل التي تمنحها Servlet، ولايريد بنفس الوقت العمل باستخدام لغة جافا.

تعتبر JSP تقنية مقادة بالصفحات أي أنه يمكن إدراج منطق العمل المطلوب ضمن سياق لغة التأشير المستخدمة. فالأمور هنا معكوسة، إذ عوضاً عن إدراج لغة التأشير ضمن لغة Java في Javaservlet يتم إدراج نصوص برمجية ضمن لغة التأشير.

يمكن لمحتويات صفحات JSP أن تُصنَّف إلى تصنيفين أساسيين:

- **العناصر:** هي عبارة عن تأشيرات شبيهة بـــ XML تُستخدم في السيطرة على سير البرنامج وتحتوي منطق جافا. وهناك خمسة أنواع لهذه العناصر هي:
  - الموجهات : أحد الموجهات هوموجه الصفحة مثلاً:

```
<%@page import="java.util.*" %>
```

- التصريحات : كما في حالة التصريح عن متحول من نوع Date في هذا المثال

```
Date theDate = new Date();
```

- التعبيرات: هي عبارة عن إدراج مباشر لقيمة ضمن خرج صفحة JSP

```
<%= new java.util.Date() %>
```

- النصيصات : تعمل بشكل مشابه للتعبيرات ولكنها تستخدم في البنى الأكثر تعقيداً

```
<%
String queryData = request.getQueryString();
out.println("Attached GET data: " + queryData);
%>
```

- الأفعال: تستخدم بنية XML للتحكم بمحرك للقيام بأفعال كإدراج نص في صفحة أوتحويل طلب صفحة إلى أخرى وأفعال أخرى.

```
<jsp:useBean id="inventoryBean" class="sample.InventoryData" />
```

- <u>**معلومات القوالب:**</u> هي أي شيء آخر ما عدا العناصر التي تم ذكرها. وعادة ما يهمل محرك JSP هذه المعلومات ويكتفي بتمريرها كما هي.

مثال:

فيما يلي مثال يوضح صفحة JSP تقوم بتوليد خرج بشكل لغة التأشير HDML

```
1.<%@ page contentType="text/x-hdml"%>
2.  <%@ page language="java"%>
3.  <!-- string declaration -->
4.  <%! String item1_id="101"; %>
5.  <%! String item2_id="102"; %>
6.  <%! String item3_id="103"; %>
7.  <!-- HDML code to display Inventory list -->
8.  <HDML VERSION="3.0">
9.    <display name="item1">
10.     <action type="accept" task="go" dest="#item2" label="Skip">
11.     <action type="soft1" task="go"
dest="details.jsp?product_id=<%=item1_id %>" label="Details">
12.     Sony-TRV30 Digital Video Camcorder
13.   </display>
14.   <display name="item2">
15.     <action type="accept" task="go" dest="#item3" label="Skip">
16.     <action type="soft1" task="go"
dest="details.jsp?product_id=<%=item2_id %>" label="Details">
17.     Hitachi-VMD875L Digital 8 Camcorder
18.   </display>
19.   <display name="item3">
20.     <action type="accept" task="go" dest="#finish" label=" finish">
21.     <action type="soft1" task="go"
dest="details.jsp?product_id=<%=item3_id %>" label="Details">
22.     Sony-DCR-IP7BT Micro MV Network Handycam
23.   </display>
```

```
24.   <display name="finish">
25.     <action type="accept" task="return" label="Done">
26.     <!-- Java scriptlet -->
28.     <%
29.         String username = request.getParameter("user");
30.         out.println("Thank-you for visiting "+ username);
31.     %>
32.   </display>
33. </HDML>
```

تُعتبر JSP الحل السحري للمطور الذي يريد الحصول على ميزة الاستقلال عن منصة العمل التي تمنحها Servlet، ولايريد بنفس الوقت العمل باستخدام لغة جافا.

تعتبر JSP تقنية مقادة بالصفحات أي أنه يمكن إدراج منطق العمل المطلوب ضمن سياق لغة التأشير المستخدمة. فالأمور هنا معكوسة، إذ عوضاً عن إدراج لغة التأشير ضمن لغة Java في Javaservlet يتم إدراج نصوص برمجية ضمن لغة التأشير.

يمكن لمحتويات صفحات JSP أن تُصنَّف إلى تصنيفين أساسيين:

- **<u>العناصر:</u>** هي عبارة عن تأشيرات شبيهة بـــ XML تُستخدم في السيطرة على سير البرنامج وتحتوي منطق جافا. وهناك خمسة أنواع لهذه العناصر هي:
  - الموجهات
  - التصريحات
  - التعبيرات
  - النصيصات
  - الأفعال

- **<u>معلومات القوالب:</u>** هي أي شيء آخر ما عدا العناصر التي تم ذكرها. وعادة ما يهمل محرك JSP هذه المعلومات ويكتفي بتمريرها كما هي.

صفحات المخدم ASP

تمتلك هذه التقنية قدرات مشابهة لتلك التي تقدمها لغة JSP فهي تقوم بالدمج بين لغة التأشير والنصوص البرمجية والمكونات من طرف المخدم في ملف واحد يعطى اللاحقة ASP.

لهذه التقنية فعالية أكبر من استخدام CGI وهي قادرة على العمل على مخدم IIS وتدعم بيئة المسالك التنفيذية المتعددة.

تم استخدام هذه التقنية قبل استخدام تقنية JSP وهي أوسع انتشاراً ويمكن استخدامها مع نوعين من اللغات Java script وVBscript.

مثال:

فيما يلي نعيد نفس المثال للنص البرمجي المستخدم في JSP ولكن هذه المرة باستخدام ASP.

```
1.  <% response.ContentType = "text/vnd.wap.wml" %>
2.  <%@ Language=VBScript %>
3.  <!-- variable declaration -->
4.  <% Item1_id="101" %>
5.  <% Item2_id="102" %>
6.  <% Item3_id="103" %>
7.  <?xml version="1.0" encoding=" UTF-8"?>
8.  <!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.1//EN"
"http://www.wapforum.org/DTD/wml_1.1.xml">
9.  <wml>
10.   <card id="item1">
11.     <do type="accept" label="next">
12.       <go href="#item2"/>
13.     </do>
14.     <do type="cancel" label="details">
15.       <go href=" details.asp?product_id=<%=Item1_id %>"/>
16.     </do>
17.     <p align="center"><b>Inventory Items</b></p>
18.     <p> Sony-TRV30 Digital Video Camcorder </p>
19.   </card>
20.   <card id="item2">
21.     <do type="accept" label="next">
22.       <go href="#item3"/>
23.     </do>
24.     <do type="cancel" label="details">
25.       <go href="details.asp?product_id=<%=Item2_id %>"/>
26.     </do>
27.     <p align="center"><b>Inventory Items</b></p>
28.     <p> Hitachi-VMD875L Digital 8 Camcorder </p>
29.   </card>
30.   <card id="item3">
31.     <do type="accept" label="next">
32.       <go href="#finish"/>
33.     </do>
34.     <do type="cancel" label="details">
35.       <go href="details.asp?product_id=<%=Item3_id %>"/>
36.     </do>
37.     <p align="center"><b>Inventory Items</b></p>
38.     <p> Sony-DCR-IP7BT Micro MV Network Handycam </p>
39.   </card>
40.   <card id="finish">
41.     <do type="accept" label="start over">
42.       <go href="#item1"/>
43.     </do>
```

```
44.     <!-- VBScript to get URL parameter -->
45.     <% userName = request.queryString("user") %>
46.     <p> Thank-you for visiting <%= userName %> </p>
47.   </card>
48. </wml>
```

ليست هذه النسخة هي النسخة الأخيرة لهذه التقنية فقط ظهر إصدار جديد لهذه التقنية يعتمد إطار العمل.NET وهويقدم العديد من المميزات التي تشمل الأداء والمعيارية.
قد تكون هذه التقنية الاختيارالأنسب في حالة اعتماد بيئة Windows كونها تملك أعلى درجة من التكاملية مع نظام التشغيل ومخدم الوب وبرمجيات Microsoft الأخرى.

تمتلك هذه التقنية قدرات مشابهة لتلك التي تقدمها لغة JSP فهي تقوم بالدمج بين لغة التأشير والنصوص البرمجية والمكونات من طرف المخدم في ملف واحد يعطى اللاحقة.ASP

لهذه التقنية فعالية أكبر من استخدام CGI وهي قادرة على العمل على مخدم IIS وتدعم بيئة المسالك التنفيذية المتعددة.

تم استخدام هذه التقنية قبل استخدام تقنية JSP وهي أوسع انتشاراً ويمكن استخدامها مع نوعين من اللغات Java script

XML واستخدام صفحات الأنماط XSL

تقنية توليد المحتوى الأخيرة التي سنتعرف عليها في هذه الجلسة هي تقنية تحويل بيانات لغة التأشير XML باستخدام ملفات صفحات الأنماط XSL.

نلاحظ في التطور من تقنيات CGI وjavaservlet إلى تقنيات كـ ASP وJSP أن أحد الأهداف لعملية التطوير تلك كانت فصل البيانات عن آلية إظهارها قدر الإمكان.

تحقق صفحات XML وXSL هذه الهدف بكفاءة عالية إذا تساعد على فصل كامل بين التقديم والبيانات حيث يتفوق هذا الحل على حلول ASP وJSP في هذا المجال.

فعلى سبيل المثال إذا كان لدينا بيانات في صفحة XML ونود الحصول على خرج لهذه البيانات بلغات تأشير مختلفة WML CHTMLو XHTML يمكننا ببساطة اللجوء إلى إنشاء صفحة XSL خاصة بكل واحدة من هذه اللغات.

مثال:

فيما يلي مثال عن ملف XML حيث سنقوم بتقديم بيانات هذا الملف باستخدام ملف XSL.

```xml
<?xml version="1.0"?>
<inventory>

  <product id="101">
    <name>
      <manufacturer>Sony</manufacturer>
      <model>TRV30</model>
    </name>
    <description>Digital Video Camcorder</description>
    <digitalstill>1360 x 1020</digitalstill>
    <format>Mini DV</format>
    <quantity>17</quantity>
    <price>1699.00</price>
  </product>

  <product id="102">
    <name>
      <manufacturer>Hitachi</manufacturer>
      <model>VMD875L</model>
    </name>
    <description>Digital 8 Camcorder</description>
    <format>Digital8</format>
    <quantity>24</quantity>
    <price>599.00</price>
  </product>

  <product id="103">
    <name>
      <manufacturer>Sony</manufacturer>
      <model>DCR-IP7BT</model>
    </name>
    <description>Micro MV Network Handycam</description>
    <digitalstill>640 x 480</digitalstill>
    <format>Micro MV</format>
    <quantity>11</quantity>
    <price>2199.99</price>
  </product>
<product id="104">
    <name>
      <manufacturer>JVC</manufacturer>
      <model>GR-DV2000</model>
    </name>
    <description>High-Band Digital Video Camcorder</description>
    <digitalstill>1600 x 1200</digitalstill>
    <format>Mini DV</format>
    <quantity>4</quantity>
    <price>1599.00</price>
  </product>

  <product id="105">
    <name>
      <manufacturer>Canon</manufacturer>
```

```
      <model>ES8200V</model>
    </name>
    <description>8 MM Camcorder</description>
    <format>HI8MM</format>
    <quantity>37</quantity>
    <price>399.00</price>
  </product>

</inventory>
```

يمثل هذا الملف معلومات مستودع يحتوي مجموعة من المنتجات لكل منها وصف، سعر، كمية..إلخ

سنحاول الآن كتابة ملف XSL لتشكيل عملية التقديم للبيانات الواردة في ملف XML.

لا بد إضافة السطر الخاص بربط ملف XML بملف XSL المطلوب لنحصل على النتيجة المطلوبة وتتم هذه العملية بإضافة هذه الصيغة إلى بداية ملف XML :

```
<?xml-stylesheet href="inventory.xsl" type="text/css"?>
```

فيما يلي سرد لمحتوى ملف XSL الذي سيقوم بتحويل خرج بيانات ملف XML إلى لغة التأشير WML

```
1.  <?xml version="1.0"?>
2.  <xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
3.  <xsl:output method="xml" indent="yes" doctype-
system=" http://www.wapforum.org/DTD/wml_1.1.xml" doctype-public="-
//WAPFORUM//DTD WML 1.1//EN" />
```

بدأنا أولاً بتعريف نوع الوثيقة وسننتقل إلى تعريف قالب سيوضح تشكيل البيانات التي سيتم استخراجها من ملف XML

```
4.  <xsl:template match="inventory">
5.  <wml>
6.    <card id="inventory">
7.      <p align="center">Inventory Items</p>
8.      <p>
9.        <select name="productId" multiple="false">
10.         <xsl:apply-templates select="product"/>
11.       </select>
12.     </p>
13.   </card>
14. </wml>
15. </xsl:template>
```

في هذه النقطة ينتهي القالب الخاص بإظهار معلومات المستودع كما نلاحظ قمنا باستخدام تعبير XSL الخاص بإظها معلومات العقد المحددة وهي في حالتنا Product

```
<xsl:apply-templates select="product"/>
```

الجزء التالي في هذا المثال سيحدد كيفية إظهار معلومات العقدة Product وذلك باستخدام تعبير XSL

```
<xsl:template match="product">
```

في حين يقوم التعبير

```
<xsl:variable name="product_id">
```

بتعريف متحول باسم Product_id ويضع ضمنه القيمة id التي يقوم باستيرادها من ملف XML بالتعبير

```
<xsl:apply-templates select="@id" />
```

```
16. <xsl:template match="product">
17.     <xsl:variable name="product_id">
18.       <xsl:apply-templates select="@id" />
19.     </xsl:variable>
20.     <option value="{$product_id}">
21.       <xsl:apply-templates select="name"/>
22.       <onevent type="onpick">
23.         <go href="details.wml">
24.           <postfield name="product_id" value="{$product_id}"/>
25.         </go>
26.       </onevent>
27.    </option>
28. </xsl:template>
29. <xsl:template match="name">
30.     <xsl:value-of select="manufacturer"/>-<xsl:value-of
select="model"/>
31. </xsl:template>
32. </xsl:stylesheet>
```

تقنية توليد المحتوى الأخيرة التي سنتعرف عليها في هذه الجلسة هي تقنية تحويل بيانات لغة التأشير XML باستخدام ملفات صفحات الأنماط XSL.

نلاحظ في التطور من تقنيات CGI وjavaservlet إلى تقنيات كـ ASP وJSP أن أحد الأهداف لعملية التطوير تلك كانت فصل البيانات عن آلية إظهارها قدر الإمكان.

تحقق صفحات XML وXSL هذه الهدف بكفاءة عالية إذا تساعد على فصل كامل بين التقديم والبيانات حيث يتفوق هذا الحل على حلول ASP وJSP في هذا المجال.

فعلى سبيل المثال إذا كان لدينا بيانات في صفحة XML ونود الحصول على خرج لهذه البيانات بلغات تأشير مختلفة WML CHTML وXHTML يمكننا ببساطة اللجوء إلى إنشاء صفحة XSL خاصة بكل واحدة من هذه اللغات.

القسم السابع والثامن:

عناصر تحكم نماذج الوب المحمولة

الكلمات المفتاحية:

عنصر تحكم، محمول، لغة تأشير، تأشيرة، واصفة، خاصة، حدث.

ملخص:

سنتعرف في هذه الجلسة على عناصر تحكم نماذج الوب المحمولة. ستغطي هذه الجلسة العناصر الحاوية، الأساسية، وتلك المستخدمة للأغراض الخاصة.

أهداف تعليمية:

يتعرف الطالب في هذا الفصل على:

- عناصر تحكم نماذج الوب الحاوية
- عناصر تحكم نماذج الوب الأساسية
- عناصر تحكم نماذج الوب ذات الأغراض الخاصة.

حل مشكلة تطوير الحلول المحمولة

شاهدنا في الجلسة الماضية العديد من التقنيات واللغات المستخدمة لتطوير الحلول المحمولة.
ولاحظنا أنه يتوجب على المطور أقلمة المحتوى لينسجم مع التنوع الكبير للتجهيزات ومواصفاتها من حيث مساحة شاشة العرض، والألوان، والمستعرض الصغري المستخدم، وبالتالي اللجوء إلى جملة من لغات التأشير كـــHTML، WML، CHTML، HDML.... الخ.

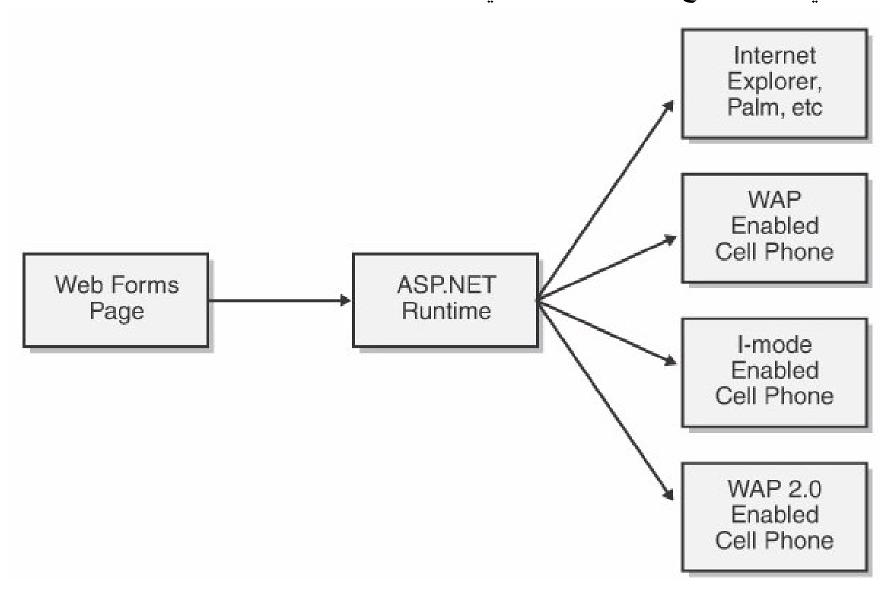
مع كل هذا الإرباك كيف يمكن للمطور أن يقوم بالعمل على كتابة تطبيقات محمولة بإنتاجية معقولة؟

تكمن الإجابة عن هذا السؤال في محاولة جعل التقنية المستخدمة من طرف الزبون غير ذات أهمية، واستخدام كيان وسيط على مخدم الوب ليتولى الاهتمام بالمتطلبات الخاصة لكل جهاز زبون بالنيابة عن الجهاز نفسه.

قدمت ASP.NET حلاً معقولاً يترجم هذه الإجابة، إذ يكفي أن يجري تطوير تطبيق باستخدام asp.net على مخدم وب IIS ليتولى إطار العمل.NET ضمان تشغيل التطبيق المطوَّر على أكثرمن 200 نوع من التجهيزات المحمولة من شركات مختلفة وبمواصفات مختلفة كلُ باستخدام لغة التأشير الخاصة به وحجم الشاشة المناسب.

لضمان استمرار صلاحية هذا الإطار تقوم Microsoft بتحديثه دورياً ليشمل التجهيزات المتوفرة في الأسواق.

فيما يلي شكل يوضح الآلية التجريدية التي تعمل فيها هذه التقنية.



التطوير باستخدام نماذج الوب المحمولة

تُستخدَم نماذج الوب المحمولة لبناء صفحات بغض النظر عن لغة التأشير التي تستخدمها التجهيزات المحمولة للزبائن.

يعمل المطور هنا على الواجهة المجردة باستخدام أغراض تمثل المكونات الأساسية التي سيجري إظهارها مثل التأشيرات النصية، ومربعات الإدخال. في حين تقع على عاتق بيئة التشغيل أخذ هذه المكونات المجردة وتحويلها إلى لغة التأشير الخاصة التي يدعمها الجهاز المحمول.

تكون أغراض نماذج الوب المحمولة عبارة عن أغراض برمجية يجري التحكم بتوضعها من خلال استخدام ملف نصي مكتوب بلغة XML يقدم هذه الأغراض.
توفر ASP.NET نماذج وب محمولة مشابهة للنماذج القياسية لتمثيل لكل عنصر من عناصر واجهة المستخدم.

مثال:
فيما يلي مثال بسيط يبرز كيفية إظهار عنصر نموذج وب محمول من نوع label.

```
<%@ Page Language="vb"
Inherits="System.Web.UI.MobileControls.MobilePage" %>
<%@ Register TagPrefix="mobile"
Namespace="System.Web.UI.MobileControls"
    Assembly="System.Web.Mobile" %>

<mobile:Form id="Form1" runat="server">
    <mobile:Label id="Label1" runat="server">Hello,
World</mobile:Label>
</mobile:Form>
```

استخدمنا في المثال موجه Page لتحديد اللغة المستخدمة واستخدام الصنف MobilePage لتوريث الصفحة الحالية. ثم سجلنا mobile لنتمكن من استخدامها في تتمة الرماز لتشكل سابقة لأسماء عناصر التحكم المحمولة المستخدمة.
تجري ترجمة هذه الصفحة عند استدعائها لأول مرة ويجري بناء التطبيق ووضعه في الذاكرة الخبيئة بغرض تخديم الطلبات التالية لنفس الصفحة.
عندما يعمل التطبيق يقوم بفحص ترويسة HTTP وتحديد نوع الجهاز الذي قام بإرسال الطلب وبالتالي نوع لغة التأشير المطلوبة.
بعدها يقوم التطبيق بتوليد لغة التأشير المناسبة تجاوباً مع الطلب.

استخدام نماذج الوب المحمولة

تعتبر عناصر التحكم المحمولة نمط خاص من عناصر تحكم نماذج الوب العادية.
يمكننا تقسيم هذه العناصر إلى أصناف نوردها فيما يلي:

عناصر التحكم الأساسية: هي عبارة عن نسخة من عناصر التحكم المعيارية مثل مربع النص، والوسوم، والصور، وغيرها، خاصة بالتطبيقات المحمولة.

عناصر تحكم التحقق: تؤمن عملية التحقق من الإدخال بما يشمل نمط البيانات، وتنسيقها أوحتى وجودها.

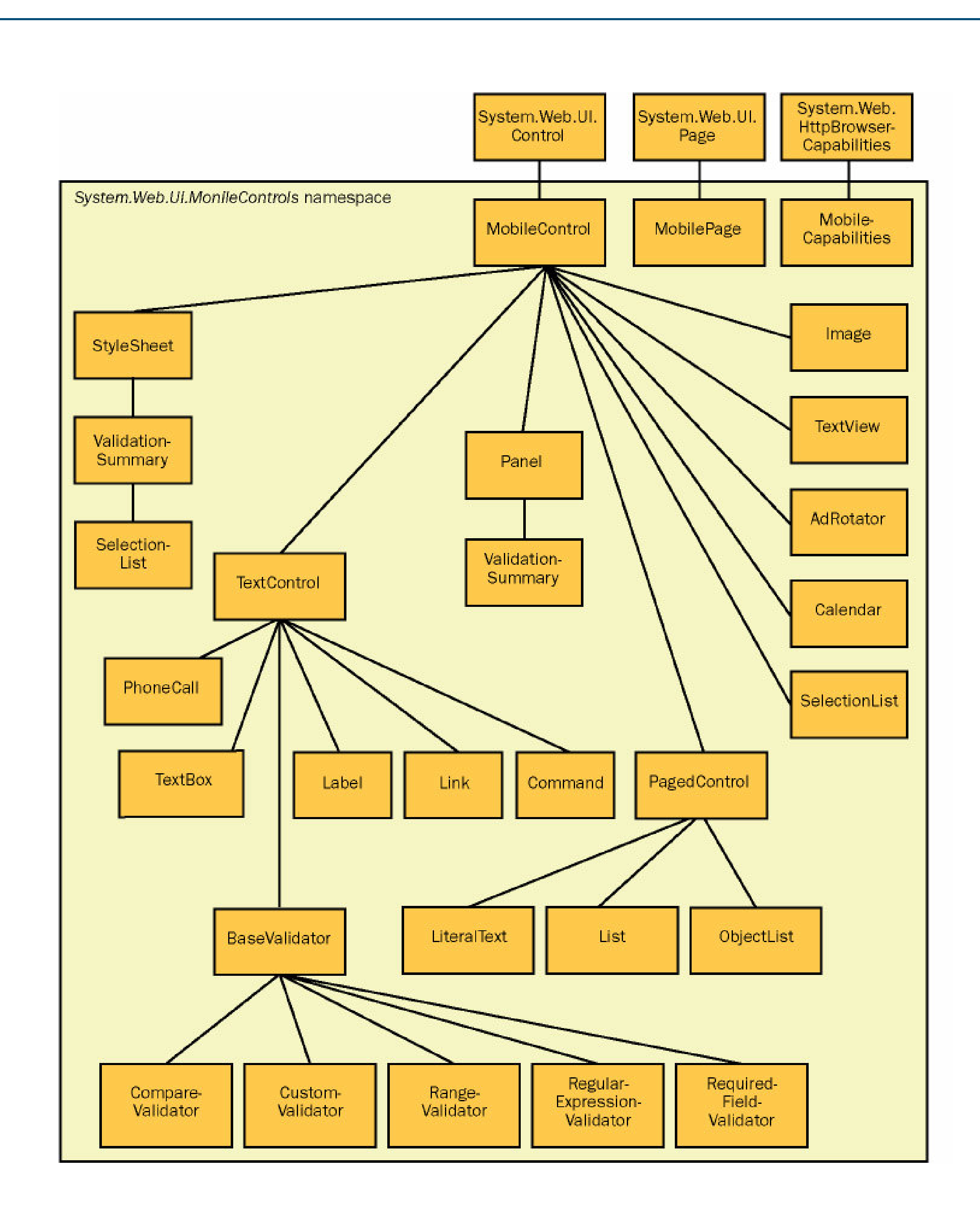
عناصر التحكم الخاصة بالزبون المحمول فقط: تقدم هذه العناصر وظائف يمكن تطبيقها فقط على الزبون المحمول مثل عنصر التحكم الخاص بتأسيس مكالمة هاتفية.

عناصر التحكم الخاصة ببنية جهاز مختار محدد: تسمح هذه العناصر بعملية ضبط التطبيق لجهاز معين عبر إعطاء قيمة معينة مثلاً لأحد خصائص عنصر التحكم من أجل نوع معين من الأجهزة.

القوالب: تدعم عناصر التحكم من نوع Form، وPanel، وList، وObjectListControls ما ندعوه بالقوالب (Template) حيث تستخدم عناصر التحكم تلك عناصر التحكم الخاصة ببنية جهاز محدد لربط قالب محتوى معين بهذا الجهاز.

البرمجة باستخدام عناصر التحكم المعيارية

قبل البدء باستعراض أهم عناصر التحكم المحمولة المعيارية نوضح بشكل بياني السوية العليا للبنية الهرمية الخاصة بعناصر التحكم المحمولة.



الخصائص العامة لعناصر التحكم المحمولة

تتحدر أغلب صفوف عناصر تحكم نماذج الوب المحمولة وترث أغلب خصائصها من الصف System.Web.UI.MobileControl الذي يرث بدوره الصف System.Web.UI.Control.

تحوي جميع عناصر التحكم المحمولة الغرض System.Web.UI.MobileControls.Style حيث لا يمكن الوصول إلى هذا

الغرض مباشرةً بل عن طريق خصائص عامة تشير بشكل ضمني إلى قيم خصائص style، فعلى سبيل المثال يحوي كل عنصر تحكم محمول الخاصة Font والتي تشير إلى خاصة داخلية محتواة في الغرض Style بحيث يمكنك تثبيت أواستخدام قيمة الخاصة الداخلية أثناء كتابة البرنامج باستخدام تعبير من الشكل Font.Italic مثلاً.

على أي حال يمكن تعيين قيم هذه الخصائص مباشرةً عن طريق واصفات عنصر التحكم من جهة المخدم بحيث يمكننا كتابة صيغة من الشكل:

```
<mobile:aMobileControl
    runat="server"
    id="id"
    BreakAfter=="{True|False}"
    Font-Name="fontName"
    Font-Size="{NotSet|Normal|Small|Large}"
    Font-Bold="{NotSet|False|True}"
    Font-Italic="{NotSet|False|True}"
    ForeColor="foregroundColor"
    BackColor="backgroundColor"
    Alignment="{NotSet|Left|Center|Right}"
    StyleReference="styleReference"
    Visible="{True|False}"
    Wrapping="{NotSet|Wrap|NoWrap}"

    <!-- Events -->
    OnDataBinding="EventHandlerMethodName"
    OnDisposed="EventHandlerMethodName"
    OnInit="EventHandlerMethodName"
    OnLoad="EventHandlerMethodName"
    OnPreRender="EventHandlerMethodName"
    OnUnLoad="EventHandlerMethodName"
</ mobile aMobileControl >
```

سنشرح في الجدول التالي كل من هذه الخصائص باختصار:

الخاصة	القيم	الوصف
Alignment	Alignment.NotSet|Left|Center|Right	تحدد اتجاه المحاذاة للعنصر ضمن الحاوية الحالية، وفي حال عدم تعيينها تأخذ المحاذاة المحددة في الحاوية أوالاتجاه الأيسر في حال عدم تعيين قيمة للحاوية
BackColor	None|hexadecimal RGB values|standard HTML color identifiers|color constants	لون الخلفية المستخدم لعنصر التحكم. القيمة الافتراضية هي Color.Empty
BreakAfter	True|False	تحدد إضافة فاصل سطر بعد عنصر التحكم. تكون القيمة

		الافتراضية هي True
Font.Size	FontSize.NotSet| Normal| Small| Large	يجري تصغير الخط باستخدام الحجم المحدد حسب مقدرات الجهاز الزبون.
Font.Bold	BooleanOption.NotSet|False|True	تحدد فيما إذا كان النص سميك أم لا.
Font.Italic	BooleanOption.NotSet|False|True	تحدد كون الخط مائل.
ForeColor	None|hexadecimal RGB values|standard HTML color identifiers|color constants	تحدد اللون المستخدم للنص في عنصر التحكم. في حال تم استخدام None يرث النص في العنصر هذه الخاصة من العنصر الذي يحويه.
ID	String	تستخدم هذه الواصفة في حال إعطائها قيمة كمرجع إلى عنصر التحكم الذي استُخدِمَت فيه. في حال لم يجر إعطاء قيمة لهذه الخاصة يقوم النظام تلقائياً بوضع قيمة تلقائية.
UniqueID	قيمة يتم تعيينها من قبل النظام	لا يمكن التصريح عن قيمة هذه الخاصة، وتقع على عاتق النظام مهمة تحديد قيمة لها. تتكون هذه القيمة من قيمة الخاصة التابعة للعنصر ID تليها قيم الخاصة ID لأي عنصر يحتوي هذا العنصر.
Visible	True|False	في حال تم إسناد القيمة False إلى هذه الخاصة يبقى العنصر موجوداً على الصفحة كغرض برمجي.
Wrapping	Wrapping. NotSet|Wrap| NoWrap	تحدد فيما إذا كان النص سيلتف عند تخطيه حدود الصفحة أم لا.

تتحدر أغلب صفوف عناصر تحكم نماذج الوب المحمولة وترث أغلب خصائصها من الصف System.Web.UI.MobileControl الذي يرث بدوره الصف System.Web.UI.Control.

تحوي جميع عناصر التحكم المحمولة الغرض System.Web.UI.MobileControls.Style حيث لا يمكن الوصول إلى هذا الغرض مباشرةً بل عن طريق خصائص عامة تشير بشكل ضمني إلى قيم خصائص style، فعلى سبيل المثال يحوي كل عنصر تحكم محمول الخاصة Font والتي تشير إلى خاصة داخلية محتواة في الغرض Style بحيث يمكنك تثبيت أواستخدام قيمة الخاصة الداخلية أثناء كتابة البرنامج باستخدام تعبير من الشكل Font.Italic مثلاً.

الأحداث العامة لعناصر التحكم المحمولة

تشترك العديد من عناصر التحكم المحمولة ببعض الأحداث وأهمها:

الحدث	الوصف
DataBinding	يظهر هذا الحدث عند ربط العنصر مع مصدر بيانات.
Disposed	يظهر هذا حدث عند تحرير مساحة الذاكرة التي يحجزها عنصر التحكم.
Init	يظهر هذا الحدث عندما يجري تأهيل عنصر التحكم وهي المرحلة الأولى في دورة حياة عنصر التحكم. في هذه المرحلة لا تكون القيمة الخاصة ViewState متوفرة كذلك لايمكن الوصول لأي قيمة من عنصر تحكم آخر من ضمن معالج هذا الحدث.
Load	يظهر لدى تحميل العنصر ضمن الغرض MobilePage. يمكن الوصول إلى قيمة ViewState وعناصر التحكم الأخرى من ضمن معالج هذا الحدث.
PreRender	يظهر هذا الحدث قبل وضع عنصر التحكم ضمن غرض MobilePage الذي يحويه.
Unload	يظهر هذا الحدث بعد تفريغ عنصر التحكم من الذاكرة.

تأخذ معالجات الأحداث لجميع هذه الأحداث معاملين الأول من النمط Object يحدد عنصر التحكم الذي قام بتوليد الحدث والثاني من نمط System.EventArgs يحدد معلومات عن الحدث نفسه.

```
EventHandlerMethodname(Object sender, EventArgs e)
```

لتحديد اسم الطريقة التي سيتم تشغيلها عند إطلاق الحدث فيمكننا استخدام الصيغة.

```
<mobile:aMobileControl
    runat="server"
    id="id"
   OnLoad="methodName"
       .
       .
       .
```

حيث يجري تشغيل الطريقة methodName عند إطلاق الحدث OnLoad.

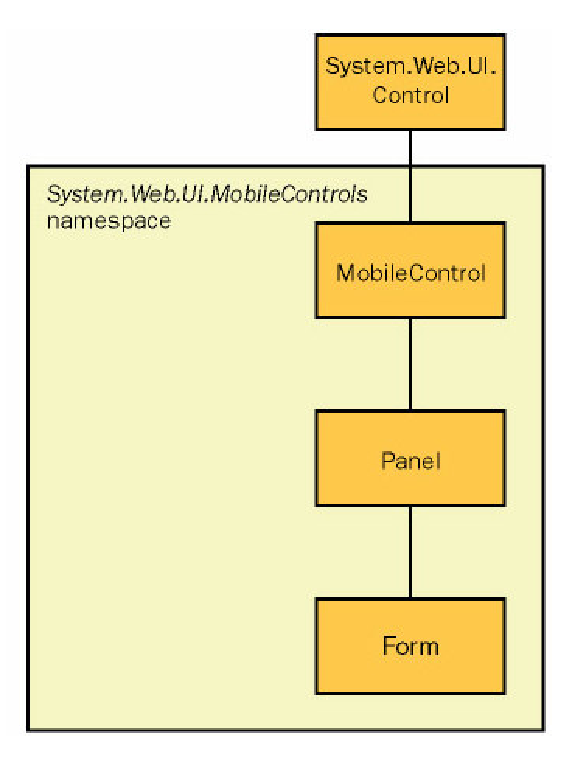
عناصر التحكم الحاوية

يطلق هذا الاسم على عناصر التحكم التي تمكننا من تجميع عناصر التحكم الأخرى. أهم تلك العناصر عنصر التحكم Form الذي يساعد في عملية تجميع عناصر التحكم الأخرى ضمن وحدات برمجية. أما العنصر الثاني فهوعنصر التحكم Panel الذي يساعد في تجميع عناصر التحكم ضمن عنصر Form الواحد.

ترث العناصر التي توضع ضمن عنصر التحكم Panel أو Form خصائص هذا العنصر المحدد للنمط Style إلا إذا تم تحديد قيم خاصة لعناصر التحكم بصورة إفرادية.

غالباً ما تمثل عناصر تحكم Panel آلية عملية لتطبيق نمط محدد على مجموعة من عناصر التحكم التي يحتويها.

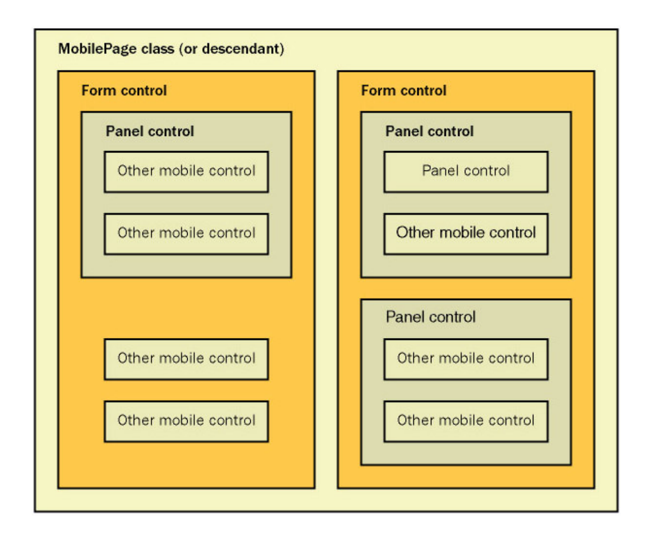
فيما يلي الشكل البياني الذي يوضح البنية الهرمية للعناصر Form وPanel:

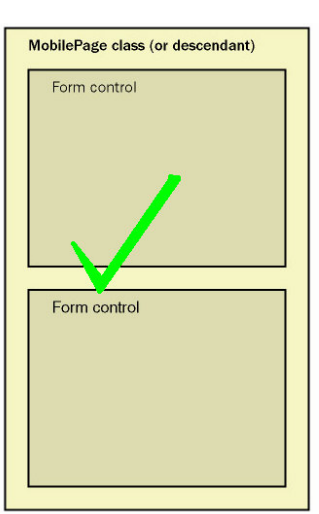


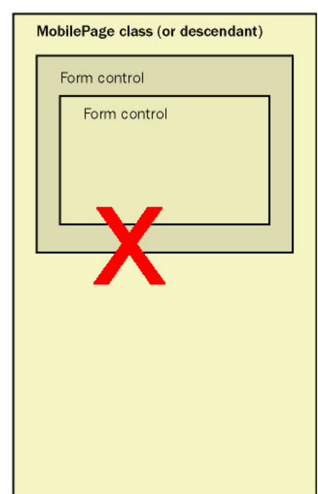
قواعد الاحتواء:

توفر عناصر التحكم الحاوية آلية فعالة لإعطاء هيكلية لتطبيق الوب، لذلك لا بد من الإلمام بشكل جيد بقواعد الاحتواء المتبادلة بين هذه العناصر وعناصر التحكم الأخرى.

للاختصار ولوضوح أكبر نورد الشكل البياني الذي يوضح قواعد الاحتواء المتبادلة بين عناصر التحكم:







عناصر التحكم الحاوية

- عنصر التحكم Form -

يعتبر عنصر التحكم Form العنصر الخارجي ضمن غرض MobilePage، حيث تحتاج صفحة نماذج الوب إلى عنصر تحكم Form واحد على الأقل لاحتواء عناصر التحكم الأخرى.

بالإضافة إلى قدرة عنصر التحكم Form على احتواء عناصر التحكم الأخرى، يستطيع عنصر التحكم Form احتواء النص خارج عناصر التحكم المحمولة حيث يمكن تنسيق هذا النص باستخدام التأشيرات الخاصة بالتنسيق مثل <p></p>،
، ، <i></i>،

ملاحظة هامة: يمكن استخدام تأشيرات التنسيق تلك فقط للنص الحر ضمن عنصر التحكم Form ولايمكن استخدامها بالنسبة للنصوص ضمن عناصر التحكم المحمولة الأخرى، أي أن الصيغة التالية لن تكون صحيحة.

```
<mobile:Label runat="server"><b>Hello World</b></mobile:Label>
```

نورد فيما يلي صيغة تحتوي معظم الخاصات المتعلقة بعنصر التحكم Form:

```
<mobile:Form
    runat="server"
    id="id"
    Font-Name="fontName"
    Font-Size="{NotSet|Normal|Small|Large}"
    Font-Bold="{NotSet|False|True}"
    Font-Italic="{NotSet|False|True}"
    ForeColor="foregroundColor"
    BackColor="backgroundColor"
    Alignment="{NotSet|Left|Center|Right}"
    StyleReference="styleReference"
    Visible="{True|False}"
    Wrapping="{NotSet|Wrap|NoWrap}"

    Action="url"
    Method="{Post|Get}"
    OnActivate="onActivateHandler"
    OnDeactivate="onDeactivateHandler"
    Paginate="{True|False}"
    PagerStyle-NextPageText="text"
    PagerStyle-PageLabel="text"
    PagerStyle-StyleReference="styleReference"
    Title="formTitle">
Child controls
</mobile:Form>
```

وسنشرح فيما يلي الخصائص التي تميز عنصر التحكم هذا:

الخاصة	النمط	الوصف
Action	String	يمثل عنوان الـــ URL الذي سيجري إرسال محتوى النموذج إليه باستخدام إحدى الطريقتين POST أو GET. في حال لم يجر تعيين قيمة لهذه الخاصة، يجري إرسال النموذج إلى نفس عنوان URL للصفحة الحالية.
ControlToPaginate	Control	تمكن هذه الخاصة من السماح لعنصر تحكم واحد على النموذج من أن يتم تحويله إلى عدة صفحات حتى وإن كانت قيمة الخاصة Paginate في عنصر تحكم النموذج تساوي القيمة False.
CurrentPage	Integer	تقوم هذه الخاصة بإعادة دليل الصفحة الحالية في حالة حدوث عملية تقسيم إلى صفحات.
Method	System.Web.UI. MobileControls. FormMethod Post | Get	تحدد الطريقة المستخدمة لإرسال طلب HTTP حيث تأخذ إحدى القيمتين Post أو Get أما القيمة الافتراضية فهي FormMethod.Post

PageCount	Integer	تعيد العدد الكلي للصفحات التي تم تقسيم النموذج إليها في حال حدوث هذه العملية.
PagerStyle	System.Web.UI. MobileControls. PagerStyle	تقوم بتحديد أوبإعادة قيمة الغرض PagerStyle الذي يحدد النص الذي سيجري إظهاره. يقوم النظام بصورة تلقائية بتوليد وصلات (Next – Previous)
Paginate	True | False	تحدد هذه القيمة المنطقية وجوب السماح بتحويل النموذج إلى أكثر من صفحة.
Title	String	يمثل عنوان النموذج ويظهر عادة في أعلى الصفحة أوفي أماكن أخرى اعتماداً على المستعرض

ملاحظة: في الحالة التلقائية لا يسمح عنصر تحكم النموذج بتقسيم الخرج إلى صفحات لذلك قد نحصل عند تجريب تطبيقاتنا على أخطاء سببها عدم دعم الجهاز لأسلوب عرض الخرج.

يعتبر عنصر التحكم Form العنصر الخارجي ضمن غرض MobilePage، حيث تحتاج صفحة نماذج الوب إلى عنصر تحكم Form واحد على الأقل لاحتواء عناصر التحكم الأخرى.

بالإضافة إلى قدرة عنصر التحكم Form على احتواء عناصر التحكم الأخرى، يستطيع عنصر التحكم Form احتواء النص خارج عناصر التحكم المحمولة حيث يمكن تنسيق هذا النص باستخدام التأشيرات الخاصة بالتنسيق

عناصر التحكم الحاوية

– عنصر التحكم Form –

يمتلك عنصر التحكم Form مجموعة من الأحداث تتضمن Activate، Deactivate، Paginate ترتبط بتنشيط وإيقاف النموذج وكذلك بعملية التقسيم إلى صفحات.

كما يدعم عنصر تحكم النموذج إمكانية الربط بالقوالب HeaderTemplate، وFooterTemplate، وScriptTemplate.

فيما يلي ندرج مثال بسيط عن كيفية استخدام عنصر تحكم النموذج المحمول.

يقوم هذا المثال بإنشاء نموذجين والسماح بالحركة بينهما باستخدام عنصر تحكم Link

```
<%@ Register TagPrefix="mobile"
    Namespace="System.Web.UI.MobileControls"
    Assembly="System.Web.Mobile" %>
<%@ Page language="c#"
    Inherits="System.Web.UI.MobileControls.MobilePage" %>

<mobile:Form id="Form1" runat="server">
```

```
    <mobile:Label id="Label1" runat="server">
        Form 1
    </mobile:Label>
    <mobile:Link id="Link1" runat="server" NavigateUrl="#Form2">
        Link
    </mobile:Link>
</mobile:Form>

<mobile:Form id="Form2" runat="server">
    <b>
        <i>Phew, you made it!</i>
    </b>
    <br>
    <mobile:Label id="Label2" runat="server">
        Form 2
    </mobile:Label>
</mobile:Form>
```

يظهر خرج هذا البرنامج على المحاكي الخاص بنوكيا على الشكل:





يمتلك عنصر التحكم Form مجموعة من الأحداث تتضمن Activate، Deactivate، Paginate ترتبط بتنشيط وإيقاف النموذج وكذلك بعملية التقسيم إلى صفحات.

كما يدعم عنصر تحكم النموذج إمكانية الربط بالقوالب HeaderTemplate، وFooterTemplate، وScriptTemplate.

عناصر التحكم الحاوية

- عنصر التحكم Panel -

لا يمتلك عنصر التحكم Panel أي خرج مرئي ولكنه يُستخدَم للتجميع المنطقي لعناصر التحكم المحمولة الأخرى (عدا عنصر تحكم النموذج). إذ يمكن لعنصر تحكم نموذج أن يحتوي عنصر تحكم Panel واحد أوأكثر، كذلك يمكن لعنصر Panel أن يحتوي عنصر تحكم Panel أو أكثر.

تجري كتابة النص البرمجي لعنصر تحكم Panel كما يلي:

```
<mobile:Panel
    runat="server"
    id="id"
    BreakAfter=="{True|False}"
    Font-Name="fontName"
    Font-Size="{NotSet|Normal|Small|Large}"
    Font-Bold="{NotSet|False|True}"
    Font-Italic="{NotSet|False|True}"
    ForeColor="foregroundColor"
    BackColor="backgroundColor"
    Alignment="{NotSet|Left|Center|Right}"
    StyleReference="styleReference"
    Visible="{True|False}"
    Wrapping="{NotSet|Wrap|NoWrap}"

    Paginate="{True|False}" >
Child controls
</mobile:Panel>
```

يجري حصر عناصر التحكم الأخرى المراد وضعها ضمن عنصر تحكم Panel هذا بين تأشيرتي:

```
<mobile:Panel></mobile:Panel>
```

يمكننا من النص البرمجي السابق الانتباه إلى أن عنصر التحكم Panel يقدم الخاصة Paginate كخاصة إضافية عمّا ذكرناه في خصائص عناصر التحكم.

تنبه هذه الخاصة محرك زمن التشغيل إلى ضرورة محاولة إبقاء العناصر المتموضعة ضمن عنصر تحكم Panel في نفس الصفحة (إن أمكن).

يُظهِر المثال التالي أربعة عناصر تحكم Label. يتوضع إثنان منهما ضمن عنصر تحكم Panel الذي يحتوي بدوره عنصر تحكم Panel آخر يحتوي عنصري تحكم Label الباقيين:

```
<%@ Register TagPrefix="mobile"
    Namespace="System.Web.UI.MobileControls"
    Assembly="System.Web.Mobile" %>
<%@ Page language="c#"
    Inherits="System.Web.UI.MobileControls.MobilePage" %>

<mobile:Form id="Form1" runat="server">
    <mobile:Panel id="Panel1"
    runat="server"
    Font-Bold="True">
        <mobile:Label id="Label1" runat="server">
            Label 1 Panel 1
        </mobile:Label>
```

```
        <mobile:Label id="Label2" runat="server">
            Label 2 Panel 1
        </mobile:Label>
        <mobile:Panel id="Panel2"
        runat="server"
        Font-Italic="True">
            <mobile:Label id="Label3" runat="server">
                 Label 1 Panel 2
            </mobile:Label>
            <mobile:Label id="Label4" runat="server">
                 Label 2 Panel 2
            </mobile:Label>
        </mobile:Panel>
    </mobile:Panel>
</mobile:Form>
```

نلاحظ هنا عملية الوراثة لنمط الخط المستخدم من عنصر التحكم Panel إلى عناصر التحكم Label المحتواة ضمنه.

يظهر خرج هذا البرنامج على المحاكي الخاص بنوكيا على الشكل:



لا يمتلك عنصر التحكم Panel أي خرج مرئي ولكنه يُستخدَم للتجميع المنطقي لعناصر التحكم المحمولة الأخرى (عدا عنصر تحكم النموذج). إذ يمكن لعنصر تحكم نموذج أن يحتوي عنصر تحكم Panel واحد أوأكثر، كذلك يمكن لعنصر Panel أن يحتوي عنصر تحكم Panel أوأكثر.

عناصر التحكم الأساسية

سنتعرف في هذه الجزء على عناصر التحكم الأساسية وهي تتضمن تلك المسؤولة عن الملاحة ضمن المستعرض كعنصر تحكم Command وعنصر تحكم Link. سنتعرف كذلك على عناصر التحكم الخاصة بالخرج مثل عناصر تحكم Label، وTextView،

و Image، بالإضافة إلى عنصر تحكم الإدخال المباشر TextBox.

عنصر التحكم Command:

يسمح عنصر التحكم هذا بعملية إرسال البيانات إلى المخدم. تختلف الطريقة التي يجري فيها إظهار عنصر التحكم هذا على منصات العمل المختلفة وهوعادة ما يظهر بشكل زر في المستعرضات التي تدعم خرج لغة التأشير HTML أويتم إظهاره كرابط في مستعرضات WML.

فيما يلي النص البرمجي الذي يوضح الخصائص والأحداث التي يزودها عنصر التحكم هذا:

```
<mobile:Command
    runat="server"
    id="id"
    Alignment="{NotSet|Left|Centre|Right}"
    BackColor="backgroundColor"
    BreakAfter=="{True|False}"
    Font-Bold="{NotSet|False|True}"
    Font-Italic="{NotSet|False|True}"
    Font-Name="fontName"
    Font-Size="{NotSet|Normal|Small|Large}"
    ForeColor="foregroundColor"
    StyleReference="StyleReference"
    Visible="{True|False}"
    Wrapping="{NotSet|Wrap|NoWrap}"

    CausesValidation="{True|False}"
    CommandArgument="commandArgument"
    CommandName="commandName"
    ImageUrl="softkeyLabel"
    OnClick="clickEventHandler"
    OnItemCommand="commandEventhandler"
    SoftkeyLabel="softkeyLabel"
    Text="Text">
TextContent
</mobile:Command>
```

يرث عنصر التحكم Command الخصائص والأحداث المشتركة من الصف MobileControl لذلك سنورد فيما يلي فقط الخصائص والأحداث الإضافية التي يقدمها عنصر التحكم هذا:

الخاصة أو الحدث	النمط	الوصف
CauseValidation	True|False	تفيد هذه الخاصة فقط في حال احتوى النموذج الذي يحتوي عنصر التحكم على عناصر تحكم خاصة بالتحقق مثل CompareValidator،CustomValidator،RangeValidator، RequieredFielValidator. عادة ما يتم إطلاق عناصر التحكم الخاصة بالتحقق عند إرسال

		البيانات بضغط زر Command ولكن في الكثير من الأحيان لا يكون هذا الأمر مطلوباً، كما هوالحال عند استخدام زر Command لإظهار لوحة خاصة باختيار التاريخ ضمن نموذج لإدخال المعلومات الشخصية.
CommandName	String	تعطي قيمة الخاصة CommandName التابعة للغرض CommandEventArgs الذي يجري تسليمه لمعالج الأحداث OnItemCommand
CommandArgument	String	تعطي قيمة الخاصة CommandArgument التابعة للغرض CommandEventArgs الذي يجري تسليمه لمعالج الأحداث OnItemCommand
Format	System.Web.UI. MobileControls. CommandFormat Button | Link	تعطي الشكل الذي سيتم تحويل شكل عنصر التحكم إليه فإما أن يأخذ شكل زر أوشكل رابط. لا تعمل هذه الخاصة إلا في حالة المستعرضات التي تدعم JavaScript
ImageUrl	String	عند تقديم عنصر التحكم Command كزر، يمكن عند تحديد هذه الخاصة أن يجري إظهار صورة زر على التجهيزات التي تدعم الصور.
الحدث Click	اسم طريقة معالج الحدث	يعطي اسم طريقة معالج الحدث. فعند ضغط المستخدم على عنصر التحكم Command تجري إعادة عنصر التحكم إلى المخدم، ويقوم محرك زمن التشغيل باستدعاء الطريقة المحددة كقيمة لهذه الخاصة.
الحدث ItemCommand	اسم طريقة معالج الحدث	كما هي الحال مع حدث Click، يحدد هذا الحدث طريقة معالج الحدث. يمكن في هذا الحدث، إدراج أية قيمة مرغوبة ضمن العامل CommandEventArgs وذلك باستخدام الخاصتين CommandName و CommandArgument
SoftKeyLabel	String	تقدم بعض التجهيزات المحمولة كالهواتف المحمولة إمكانية الاتستخدام بأزرار برمجية، تسمح هذه الخاصة بتغيير النص الذي يظهر ضمن هذه الأزرار. تكون القيمة التلقائية للأزرار البرمجية هي GO.
Text	String	يمكن تحديد النص الذي سيجري إظهاره لعنصر التحكم هذا إما بوضع هذا النص ضمن التأشيرة الخاصة بعنصر التحكم أوبإسناد القيمة المطلوبة.

يوضح المثال البسيط التالي عمل عنصر التحكم هذا:

```
<%@ Register TagPrefix="mobile"
Namespace="System.Web.UI.MobileControls"
    Assembly="System.Web.Mobile" %>
<%@ Page language="c#" Codebehind="CommandExample.aspx.cs"
    Inherits="MSPress.MobWeb.CmdEx.MyWebForm" %>

<mobile:Form id="Form1" runat="server">
<mobile:Command id="Command1" runat="server" CommandName="RED"
    OnItemCommand="Command_SelectEvent" BackColor="Red">
    Red
</mobile:Command>
<mobile:Command id="Command2" runat="server" CommandName="BLUE"
    OnItemCommand="Command_SelectEvent" BackColor="Blue"
ForeColor="White">
    Blue
</mobile:Command>
<mobile:Command id="Command3" runat="server" CommandName="GREEN"
    OnItemCommand="Command_SelectEvent" BackColor="Lime">
    Green
</mobile:Command>
<mobile:Label id="Message" runat="server"></mobile:Label>
</mobile:Form>
```

وفيما يلي النص البرمجي CommandExample.aspx.cs المكتوب بلغة C#:

```
using System;
using System.Web.UI.WebControls;

namespace MSPress.MobWeb.CmdEx
{
    public class MyWebForm :
        System.Web.UI.MobileControls.MobilePage
    {
        protected System.Web.UI.MobileControls.Label Message;

        protected void Command_SelectEvent(
            Object sender, CommandEventArgs e)
        {
           if(e.CommandName=="RED")
                Message.Text="You selected the Red option";
            else if(e.CommandName=="BLUE")
                Message.Text="You selected the Blue option";
            else
                // Catchall case
                Message.Text="You selected the Green option";
        }
    }
}
```

سنتعرف في هذه الجزء على عناصر التحكم الأساسية وهي تتضمن تلك المسؤولة عن الملاحة ضمن المستعرض كعنصر تحكم Command وعنصر تحكم Link. سنتعرف كذلك على عناصر التحكم الخاصة بالخرج مثل عناصر تحكم Label، وTextView، وImage، بالإضافة إلى عنصر تحكم الإدخال المباشر TextBox.

عنصر التحكم Command:

يسمح عنصر التحكم هذا بعملية إرسال البيانات إلى المخدم. تختلف الطريقة التي يجري فيها إظهار عنصر التحكم هذا على منصات العمل المختلفة وهوعادة ما يظهر بشكل زر في المستعرضات التي تدعم خرج لغة التأشير HTML أويتم إظهاره كرابط في مستعرضات WML.

عناصر التحكم الأساسية

عنصر تحكم Image:

يساعد هذا العنصر في إظهار ملفات صور. يشكل هذا العنصر مشكلة بالنسبة للمطور نظراً للاختلاف والتفاوت بين تنسيقات ملفات الصور التي تدعمها التجهيزات المحمولة المختلفة، إضافةً إلى الاختلاف بين قدرات عرضشاشات هذه التجهيزات.

لذا يجب تأمين ملف الصورة بأكثر من تنسيق وتحديد استجابة كل نوع من التجهيزات لإرسال التنسيق المناسب حسب الجهاز الذي قام بإرسال الطلب.

ملاحظة : يمكن تجنب هذه الطريقة باستخدام عنصر التحكم الخاص المسمى DymamicImage الذي يقوم تلقائياً بتحويل تنسيق الصورة إلى التنسيق المناسب حيث سنأتي على شرح عنصر التحكم هذا بالتفصيل لاحقاً.

فيما يلي النص البرمجي الذي يوضح الخصائص والأحداث التي يقدمها عنصر التحكم هذا:

```
<mobile:Image
    runat="server"
    id="id"
    Alignment="{NotSet|Left|Centre|Right}"
    BackColor="backgroundColor"
    BreakAfter=="{True|False}"
    Font-Bold="{NotSet|False|True}"
    Font-Italic="{NotSet|False|True}"
    Font-Name="fontName"
    Font-Size="{NotSet|Normal|Small|Large}"
    ForeColor="foregroundColor"
    StyleReference="StyleReference"
    Visible="{True|False}"
    Wrapping="{NotSet|Wrap|NoWrap}"

    AlternateText="AltText"
    ImageUrl="masterImageSource"
    NavigateUrl="targetURL"
    SoftkeyLabel="softkeyLabel">

Optional DeviceSpecific/Choice construct here.

</mobile:Image>
```

إضافة إلى الخصائص العامة التي سبق لنا الحديث عنها، يقدم عنصر التحكم هذا مجموعة من الخصائص الإضافية نذكر أهمها:

الخاصة	نوعها	الوصف
AlternateText	String	يحدد النص الذي سيظهر على التجهيزات التي لا تدعم إظهار الصور، عوضاً عن الصورة المستخدمة. كما تحدد مكان الصورة على المستعرض بانتظار تحميل هذه الصورة من المخدم.
ImageURL	String	تحدد مصدر ملف الصورة المطلوب استخدامها. يمكننا هنا استخدام المسار النسبي أوالمسار المطلق حسب موقع ملف الصورة المراد إظهارها.
NaviagteURL	String	إذا قمنا باستخدام هذه الخاصة تتحول الصورة إلى رابط، ويمثل العنوان NaviagteURL العنوان الذي سيتم الانتقال إليه عند الضغط على رابط الصورة. في حال بدء هذه القيمة بالإشارة (#) تجري ترجمة العنوان كقيمة ID لعنصر تحكم Form في نفس الصفحة.

من أهم أنماط ملفات الصور التي يدعمها عنصر التحكم هذا: gif، وwbmp، وjpg، وPng.

عناصر التحكم الأساسية

استخدام عنصر التحكم Image:

ذكرنا مسبقاً بأنه يمكن استخدام الصور عبر تحديد ملف الصورة الموافق للتنسيق الذي يدعمه الجهاز.

يبين المثال التالي كيفية تطبيق التأشيرة <DeviceSpecific> مع التأشيرة <choice> ضمن عنصر التحكم Image لاستخدام تنسيقات صور مختلفة بحسب دعم الجهاز لهذا التنسيق:

```
<%@ Page Inherits="System.Web.UI.MobileControls.MobilePage"
    Language="c#" %>
<%@ Register TagPrefix="mobile"
    Namespace="System.Web.UI.MobileControls"
    Assembly="System.Web.Mobile" %>

<mobile:Form runat="server">
  <mobile:Image runat="server" id="Image1"
        AlternateText="Northwind Corp.">
    <DeviceSpecific>
      <Choice Filter="isHTML32"
        ImageUrl="Northwind.gif"/>
```

```
      <Choice Filter="isWML11"
        ImageUrl="Northwind.wbmp"/>
    </DeviceSpecific>
  </mobile:Image>
</mobile:Form>
```

لا بد لإتمام هذا العمل تعريف الفلاتر المُستخدَمة ضمن ملف الإعداد Web.config والذي سيحتوي النص التالي:

```
<?xml version="1.0" encoding="utf-8" ?>
<configuration>
  <system.web>
    <deviceFilters>
        <!-- Markup Languages -->
        <filter name="isHTML32"
                compare="preferredRenderingType" argument="html32" />
        <filter name="isWML11"
                compare="preferredRenderingType" argument="wml11" />
    </deviceFilters>
 </system.web>
</configuration>
```

استخدام الصور والرموز المبيتة:

يعطي عنصر تحكم Image الخاصة ImageUrl قيمة تشير إلى رمز أوصورة مبيتة ضمن الجهاز المحمول. تُرقَّم هذه الرموز عادةً ابتداءاً من Symbol:0000 أوتأخذ في بعض أنواع التجهيزات الصيغة Symbol:X00 حيث يمكن أن يكون X أحد الأحرف G أو E أو F، أما 00 فيعبر عن رقم ست عشري.

كما يمكن استخدام اسم الرمز بالشكل Symbol:cloudy حيث cloudy هنا اسم الرمز المعبر عن الجوالغائم. يشبه المثال التالي المثال السابق في هذه الشريحة ولكنه يستخدم رمز Symbol:cloudy:

```
<%@ Register TagPrefix="mobile"
    Namespace="System.Web.UI.MobileControls"
    Assembly="System.Web.Mobile" %>
<%@ Page Inherits="System.Web.UI.MobileControls.MobilePage"
    Language="c#" %>

<mobile:Form runat="server">
    <mobile:Label runat="server">
        The Weather today will be...</mobile:Label>
    <mobile:Image runat="server"
                  AlternateText="Cloudy!"
                  ImageUrl="cloudy.jpg">
        <DeviceSpecific>
            <Choice ImageUrl="symbol:cloud" Filter="isUP4x">
            </Choice>
        </DeviceSpecific>
    </mobile:Image>
```

```
    <br>
    <mobile:Image runat="server"
                  AlternateText="GoTo MSN"
                  ImageUrl="MSNlogosmall.gif"
                  NavigateUrl="http://mobile.msn.com">
        <DeviceSpecific>
            <Choice ImageUrl="MSNlogo.gif" Filter="isPocketIE">
            </Choice>
        </DeviceSpecific>
    </mobile:Image>
</mobile:Form>
```

هناك ضرورة لاستخدام ملف Web.config لتعريف الفلاتر:

```
<?xml version="1.0" encoding="utf-8" ?>
<configuration>
  <system.web>
    <deviceFilters>
        <!-- Device Browsers -->
        <filter name="isGoAmerica"
            compare="browser" argument="Go.Web" />
        <filter name="isMME" compare="browser"
            argument="Microsoft Mobile Explorer" />
        <filter name="isMyPalm" compare="browser" argument="MyPalm" />
        <filter name="isPocketIE" compare="browser" argument="Pocket
IE" />
        <filter name="isUP3x"
            compare="type" argument="Phone.com 3.x Browser" />
        <filter name="isUP4x"
            compare="type" argument="Phone.com 4.x Browser" />
    </deviceFilters>
  </system.web>
</configuration>
```

سيكون خرج هذا البرنامج كالتالي بحسب المستعرض:







استخدام عنصر التحكم Image:

ذكرنا مسبقاً بأنه يمكن استخدام الصور عبر تحديد ملف الصورة الموافق للتنسيق الذي يدعمه الجهاز.

استخدام الصور والرموز المبيتة:

يعطي عنصر تحكم Image الخاصة ImageUrl قيمة تشير إلى رمز أوصورة مبيتة ضمن الجهاز المحمول. تُرقَّم هذه الرموز عادةً ابتداءاً من Symbol:0000 أوتأخذ في بعض أنواع التجهيزات الصيغة Symbol:X00 حيث يمكن أن يكون X أحد الأحرف G أو E أو F، أما 00 فيعبر عن رقم ست عشري.

عناصر التحكم الأساسية

عنصر التحكم Label:

يساعد عنصر التحكم هذا في إظهار نص قصير مخصص للقراءة على خرج شاشة الجهاز المحمول.

فيما يلي الصيغة المستخدمة لإدراج عنصر التحكم هذا. وقد تم فيها توضيح القيم التي تأخذها الخصائص المختلفة :

```
<mobile:Label
    runat="server"
    id="id"
    Alignment="{NotSet|Left|Centre|Right}"
    BackColor="backgroundColor"
    BreakAfter=="{True|False}"
    Font-Bold="{NotSet|False|True}"
    Font-Italic="{NotSet|False|True}"
    Font-Name="fontName"
    Font-Size="{NotSet|Normal|Small|Large}"
    ForeColor="foregroundColor"
    StyleReference="StyleReference"
    Visible="{True|False}"
    Wrapping="{NotSet|Wrap|NoWrap}"

    Text="Text">
TextContent
</mobile:Label>
```

لا يقدم عنصر التحكم سوى خاصة واحدة هي Text إضافة إلى الخصائص المشتركة الموضحة سابقاً. تساعد هذه الخاصة في تحديد قيمة إلى النص المراد إظهاره ضمن عنصر التحكم هذا. تعتبر هذه الطريقة بديلاً لوضع النص ضمن تأشيرة <mobile:Label></mobile:Label>

المثال التالي يوضح استخدام عنصر التحكم هذا :

```
<%@ Register TagPrefix="mobile"
    Namespace="System.Web.UI.MobileControls"
    Assembly="System.Web.Mobile" %>
<%@ Page Inherits="MSPress.MobWeb.LblEx.MyWebForm"
AutoEventWireup="False"
    Language="c#"  CodeBehind="LabelExample.aspx.cs" %>
```

```
<mobile:Form runat="server" id="Form1">
    <mobile:Label id="Label1" runat="server"
        StyleReference="title"
        Alignment="Center">
        Centered Title
    </mobile:Label>
    <mobile:Label id="Label2" runat="server"></mobile:Label>
</mobile:Form>
```

أما النص البرمجي الذي سيتم استخدامه في الخلفية والذي يأخذ الاسم LabelExample.aspx.cs. النص هنا يقوم بتعيين قيمة النص لعنصر التحكم Label2 عند تحميل الصفحة.

```
using System;
using System.Web.UI.MobileControls;

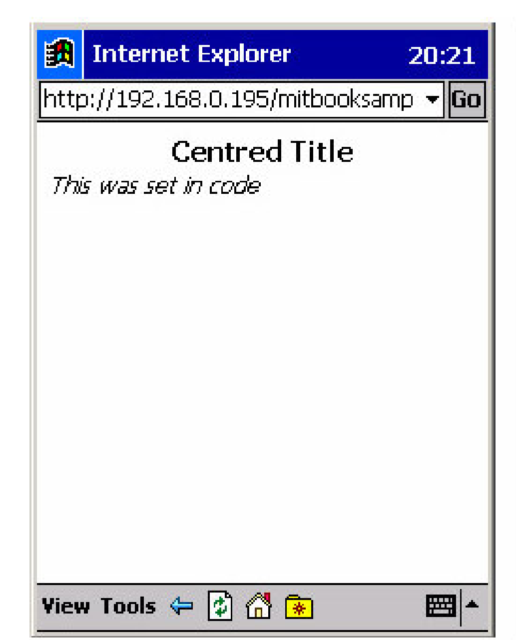
namespace MSPress.MobWeb.LblEx
{
    public class MyWebForm : System.Web.UI.MobileControls.MobilePage
    {
        protected System.Web.UI.MobileControls.Label Label2;

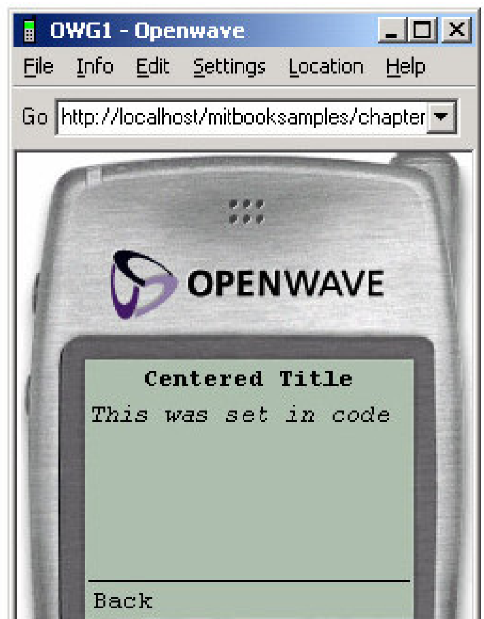
        override protected void OnInit(EventArgs e)
        {
            InitializeComponent();
            base.OnInit(e);
        }

        private void InitializeComponent()
        {
            this.Load += new System.EventHandler(this.Page_Load);
        }

        protected void Page_Load(Object sender, EventArgs e)
        {
            Label2.Text = "This was set in code";
            Label2.Font.Italic = BooleanOption.True;
        }
    }
}
```

تكون نتيجة تنفيذ هذا النص البرمجي على الشكل :





عنصر التحكم Label:

يساعد عنصر التحكم هذا في إظهار نص قصير مخصص للقراءة على خرج شاشة الجهاز المحمول.

عناصر التحكم الأساسية

عنصر التحكم Link:

يساعد هذا العنصر في إنشاء رابط تشعبي على صفحة نموذج الوب المحمول، ويقوم بالربط مع عنصر نموذج آخر ضمن نفس الصفحة أومع مصدر ما على الانترنت يُحدد بعنوان URL له.

يبين النص البرمجي التالي صيغة استخدام هذا العنصر وأهم الخصائص والأحداث التي يقدمها والقيم التي يمكن أن تأخذها.

```
<mobile:Link
    runat="server"
    id="id"
    Alignment="{NotSet|Left|Centre|Right}"
    BackColor="backgroundColor"
    BreakAfter=="{True|False}"
    Font-Bold="{NotSet|False|True}"
    Font-Italic="{NotSet|False|True}"
    Font-Name="fontName"
    Font-Size="{NotSet|Normal|Small|Large}"
    ForeColor="foregroundColor"
    StyleReference="StyleReference"
    Visible="{True|False}"
    Wrapping="{NotSet|Wrap|NoWrap}"

    NavigateUrl="target"
```

```
    SoftkeyLabel="softkeyLabel"
    Text="Text">
TextContent
</mobile:Link>
```

من النص السابق نلاحظ أن هذا العنصر يقدم ثلاث خصائص إضافية عن تلك العامة التي سبق وغطيناها.

الخاصة	النمط	الوصف
NavigateUrl	String	تعين هذه الخاصة عنوان المصدر القياسي المراد الوصول إليه.
SoftKeyLabel	String	تساعد في تغيير نص القيمة الافتراضية للزر البرمجي الذي توفره بعض المستعرضات من GO إلى أي قيمة مرغوبة أخرى.
Text	String	يمكننا تحديد النص الواجب إظهاره كنص للرابط من خلال هذه الخاصة.

يوضح المثال التالي استخدام عنصر التحكم هذا:

```
<%@ Register TagPrefix="mobile"
    Namespace="System.Web.UI.MobileControls"
    Assembly="System.Web.Mobile" %>
<%@ Page Inherits="System.Web.UI.MobileControls.MobilePage"
    Language="c#" %>

<mobile:Form runat="server" id="Form1">
    <mobile:Link id="Link1" runat="server"
                 SoftkeyLabel="->Hello"
                 NavigateURL="#Form2">
        GoTo Hello
    </mobile:Link>
    <mobile:Link id="Link2" runat="server"
                 SoftkeyLabel="->Bye"
                 NavigateURL="#Form3">
        GoTo Goodbye
    </mobile:Link>
    <mobile:Link id="Link3" runat="server"
                 StyleReference="subcommand" SoftkeyLabel="MSN"
                 NavigateURL="http://mobile.msn.com">
        MSN Mobile
    </mobile:Link>
</mobile:Form>

<mobile:Form id="Form2" runat="server">
    <B><I>Hello!</I></B>
</mobile:Form>

<mobile:Form id="Form3" runat="server">
    <B><I>Goodbye</I></B>
</mobile:Form>
```

توضح اللقطة التالية خرج هذا المثال على محاكي نوكيا ومحاكي IE





عنصر التحكم Link:

يساعد هذا العنصر في إنشاء رابط تشعبي على صفحة نموذج الوب المحمول، ويقوم بالربط مع عنصر نموذج آخر ضمن نفس الصفحة أومع مصدر ما على الانترنت يُحدد بعنوان URL له.

عناصر التحكم الأساسية

عنصر التحكم TextBox:

يسمح عنصر التحكم هذا بإدخال سطر وحيد كما يمكن لهذا العنصر أن يحتوي قيمة تلقائية مع تمكين المستخدم من تعديل هذه القيمة أو استبدالها.

فيما يلي صيغة استخدام عنصر تحكم TextBox:

```
<mobile:TextBox
    runat="server"
    id="id"
    Alignment="{NotSet|Left|Centre|Right}"
    BreakAfter=="{True|False}"
    StyleReference="StyleReference"
    Visible="{True|False}"
    Wrapping="{NotSet|Wrap|NoWrap}"

    MaxLength="maxlength"
    Numeric="{True|False}"
    Password="{True|False}"
    OnTextChanged="textChangedEventHandler"
    Size="textBoxLength"
    Text="Text"
    Title="Text"
    WmlFormat="formatMask">
TextContent
</mobile:TextBox>
```

نلاحظ أننا قمنا باستبعاد الواصفات المتعلقة بالخط ولونه والخلفية لأنه سيتم إهمالها عند تحويل هذا العنصر. أما بالنسبة للخصائص

الأخرى المختلفة عن تلك العامة فهي:

الخاصة أو الحدث	النمط	الوصف
MaxLengh	Integer	تقوم بتعيين أو إعادة الطول الأقصى المسموح لإدخال النص. تعتبر هذه القيمة غير محدودة في حال تعيين هذه الخاصة إلى القيمة 0.
Numeric	True|False	تحدد فيما إذا كان مسموح إدخال القيم الرقمية فقط.
Password	True|False	تقوم بتعيين أو إعادة القيمة المنطقية التي تحدد كون الإدخال سيتم تشفيره وتحويله إلى إشارات * أو إشارات أخرى.
Size	Integer	يحدد طول سلسلة المحارف الذي سيتم تحويل هذا العنصر إليها
Text	String	تعبر عن النص ضمن عنصر التحكم، ويمكن تحديد هذا النص بطريقتين إما بوضع النص بين فتح وإغلاق تأشيرة عنصر التحكم هذا أوبإعطاء قيمة لهذه الخاصة.
Title	String	تُهمَل هذه القيمة في الكثير من المستعرضات.
الحدث TextChanged	اسم طريقة معالج الحدث	تحدد اسم طريقة معالج الحدث. يظهر هذا الحدث عند تعديل محتوى عنصر التحكم TextBox وإرسال القيمة المعدلة إلى المخدم.
wmlFormat	String	تمكن WML من تقييد عملية الإدخال بقناع معين. فعلى سبيل المثال يحدد القناع NNNN إدخال أربع محارف رقمية.

يوضح المثال التالي عمل عنصر تحكم مربع النص TextBox:

```
<%@ Register TagPrefix="mobile"
    Namespace="System.Web.UI.MobileControls"
    Assembly="System.Web.Mobile" %>
<%@ Page Inherits="MSPress.MobWeb.TBEx.MyWebForm"
AutoEventWireup="False"
    Language="c#" CodeBehind="TextBoxExample.aspx.cs" %>

<mobile:Form runat="server" id="Form1" title="Confirm Password">
    <mobile:Label runat="server" id="Label1">
        Enter new password</mobile:Label>
    <mobile:Label runat="server" id="Label2" Visible="False"/>
    <mobile:TextBox runat="server" id="TextBox1"
                    Password="True">
    </mobile:TextBox>
    <mobile:Label runat="server" id="Label3">
        Confirm password
    </mobile:Label>
    <mobile:TextBox runat="server" id="TextBox2"
                    Password="True"/>
    <mobile:Label runat="server" id="Label4"/>
    <mobile:Command runat="server" id="cmdButton">OK</mobile:Command>
</mobile:Form>
```

أما النص البرمجي المرتبط والمحتوى في الملف TextBoxExample.aspx.cs فهوكما يلي:

```
using System;
using System.Web.UI.MobileControls;

namespace MSPress.MobWeb.TBEx
{
    public class MyWebForm : System.Web.UI.MobileControls.MobilePage
    {
        protected System.Web.UI.MobileControls.Label Label1;
        protected System.Web.UI.MobileControls.Label Label2;
        protected System.Web.UI.MobileControls.Label Label3;
        protected System.Web.UI.MobileControls.Label Label4;
        protected System.Web.UI.MobileControls.TextBox TextBox1;
        protected System.Web.UI.MobileControls.Command cmdButton;
        protected System.Web.UI.MobileControls.Form Form1;
        protected System.Web.UI.MobileControls.TextBox TextBox2;

        override protected void OnInit(EventArgs e)
        {
            InitializeComponent();
            base.OnInit(e);
        }

        private void InitializeComponent()
        {
            this.TextBox2.TextChanged +=
                new System.EventHandler(this.Verify_OnTextChanged);
            this.cmdButton.Click +=
                new System.EventHandler(this.cmdButton_Click);

        }

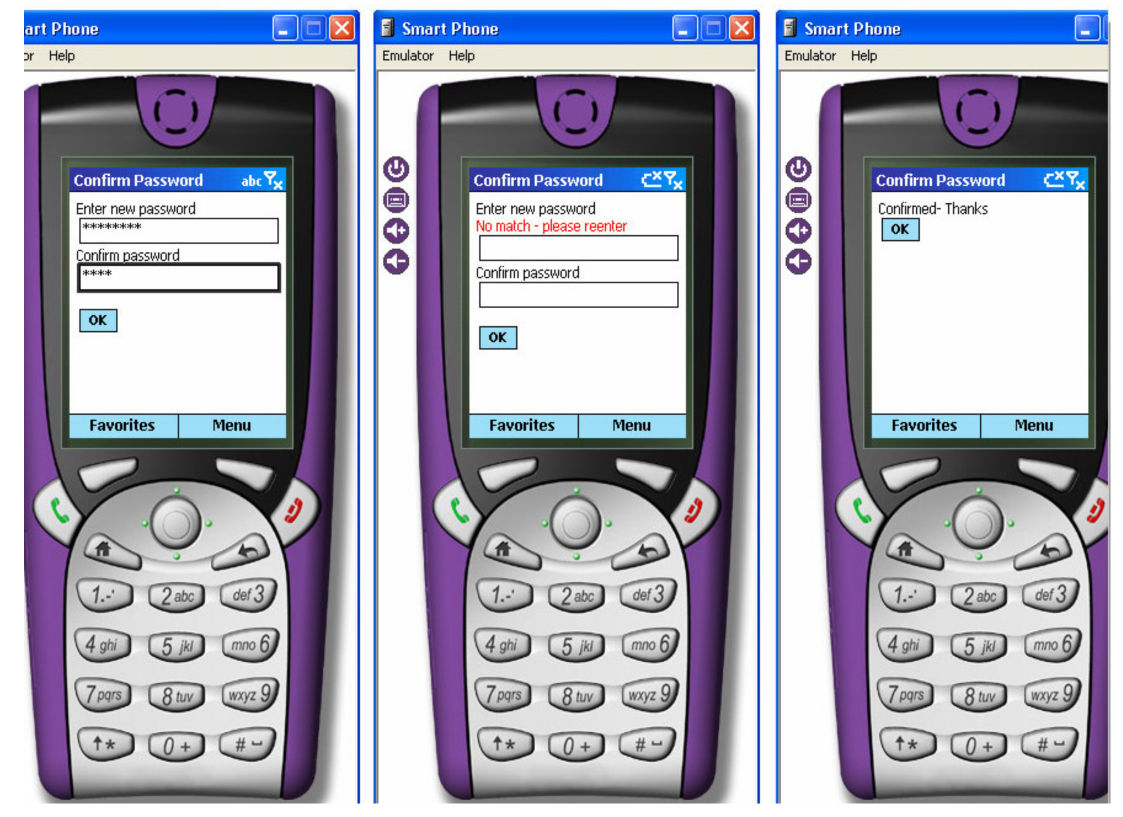
        protected void Verify_OnTextChanged(Object sender, EventArgs e)
        {
            if (TextBox1.Text != TextBox2.Text)
            {
                Label2.Visible = true;
                Label2.StyleReference = "error";
                Label2.Text = "No match - please reenter";
            }
        }

        protected void cmdButton_Click(Object sender, EventArgs e)
        {
            if (TextBox1.Text == TextBox2.Text)
            {
                Label1.Visible = false;
                Label2.Visible = false;
                Label3.Visible = false;
                TextBox1.Visible = false;
```

```
                    TextBox2.Visible = false;
                    Label4.Text = "Confirmed - Thanks";
                }
            }
        }
}
```

يقوم هذا النص البرمجي بتمكين المستخدم من إدخال كلمة سر وتأكيد صحة الكلمة بإدخالها مرة أخرى باستخدام عنصر تحكم TextBox. يقوم النص البرمجي بعد إرسال البيانات عند الضغط على زر OK الذي يمثله عنصر تحكم Command بالتحقق من التطابق بين الكلمات المدخلة وإظهار رسالة خطأ في حال عدم التطابق أوإظهار رسالة تأكيد عند التطابق.

يظهر البرنامج الخرج التالي:



عنصر التحكم TextBox:

يسمح عنصر التحكم هذا بإدخال سطر وحيد كما يمكن لهذا العنصر أن يحتوي قيمة تلقائية مع تمكين المستخدم من تعديل هذه القيمة أو استبدالها.

عناصر التحكم الأساسية

عنصر التحكم TextView:

يسمح عنصر التحكم هذا بإظهار نص طويل لايمكن إظهاره باستخدام عنصر التحكم Label.

بعكس Label يدعم هذا العنصر التقسيم الداخلي إلى صفحات بحيث يجري تقسيم النص الطويل إلى مجموعة من الصفحات بشرط تمكين هذه العملية من عنصر النموذج الذي يحوي عنصر التحكم **TextView**.

يوضح النص التالي صيغة استخدام عنصر تحكم TextView:

```
<mobile:TextView
    runat="server"
    id="id"
    Alignment="{NotSet|Left|Centre|Right}"
    BackColor="backgroundColor"
    BreakAfter=="{True|False}"
    Font-Bold="{NotSet|False|True}"
    Font-Italic="{NotSet|False|True}"
    Font-Name="fontName"
    Font-Size="{NotSet|Normal|Small|Large}"
    ForeColor="foregroundColor"
    StyleReference="StyleReference"
    Visible="{True|False}"
    Wrapping="{NotSet|Wrap|NoWrap}"

    Text="Text">
TextContent
</mobile:TextView>
```

في حالة عنصر التحكم هذا أيضاً نلاحظ وجود خاصة واحدة إضافية عمّا قمنا بتغطيته في الخصائص العامة وهي الخاصة Text حيث تحدد هذه الخاصة النص الذي سيظهر ضمن هذا العنصر.
كما هوالحال في معظم عناصر التحكم التي توفر هذه الخاصة، يمكن ضبط هذه الخاصة أيضاً عن طريق حصر النص بين تأشيرتي الفتح والإغلاق لعنصر التحكم هذا.

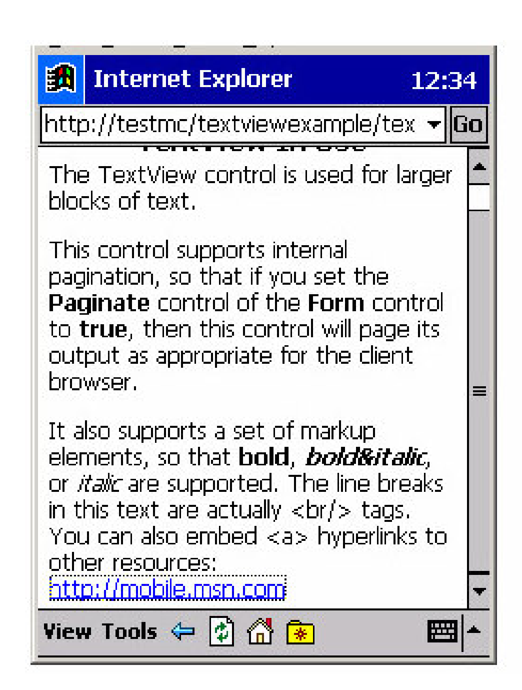
يوضح المثال التالي كيفية استخدام هذا العنصر عملياً:

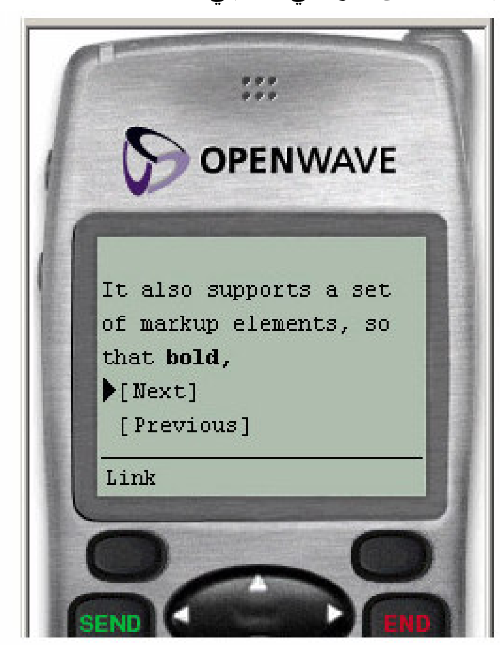
```
<%@ Page language="c#"
    Inherits="System.Web.UI.MobileControls.MobilePage" %>
<%@ Register TagPrefix="mobile"
    Namespace="System.Web.UI.MobileControls"
    Assembly="System.Web.Mobile" %>

<mobile:Form runat="server" id="Form1" Paginate="True">
    <mobile:Label id="Label1" runat="server" StyleReference="title"
                  Alignment="Center">
        TextView In Use
    </mobile:Label>
    <mobile:TextView id="TextView1" runat="server">
        The TextView control is used for larger blocks of text.
        <br />
        <br />
```

```
        This control supports internal pagination so that if you set
        the <b>Paginate</b> control of the <b>Form</b> control to
        <b>true</b>, this control will page its output as
        appropriate for the client browser.<br />
        <br />
        It also supports a set of markup elements so that <b>bold</b>,
        <b><i>bold&amp;italic</i></b>, or <i>italic </i>are supported.
        The line breaks in this text are actually &lt;br/&gt; tags.
        You can also embed &lt;a&gt; hyperlinks to other resources:
        <br />
        <a href='http://mobile.msn.com'>http://mobile.msn.com</a>
    </mobile:TextView>
</mobile:Form>
```

تظهر نتيجة تنفيذ هذا النص البرمجي كما يلي:





عنصر التحكم TextView:

يسمح عنصر التحكم هذا بإظهار نص طويل لايمكن إظهاره باستخدام عنصر التحكم Label.

بعكس Label يدعم هذا العنصر التقسيم الداخلي إلى صفحات بحيث يجري تقسيم النص الطويل إلى مجموعة من الصفحات بشرط تمكين هذه العملية من عنصر النموذج الذي يحوي عنصر التحكم **TextView**.

عناصر تحكم الأغراض الخاصة

تتضمن عناصر تحكم ASP.NET المحمولة ثلاثة عناصر تحكم ذات أغراض خاصة تشترك في إغناء تطبيقات الوب المحمولة.هذه العناصر هي Clendar، PhoneCall، AdRotator.

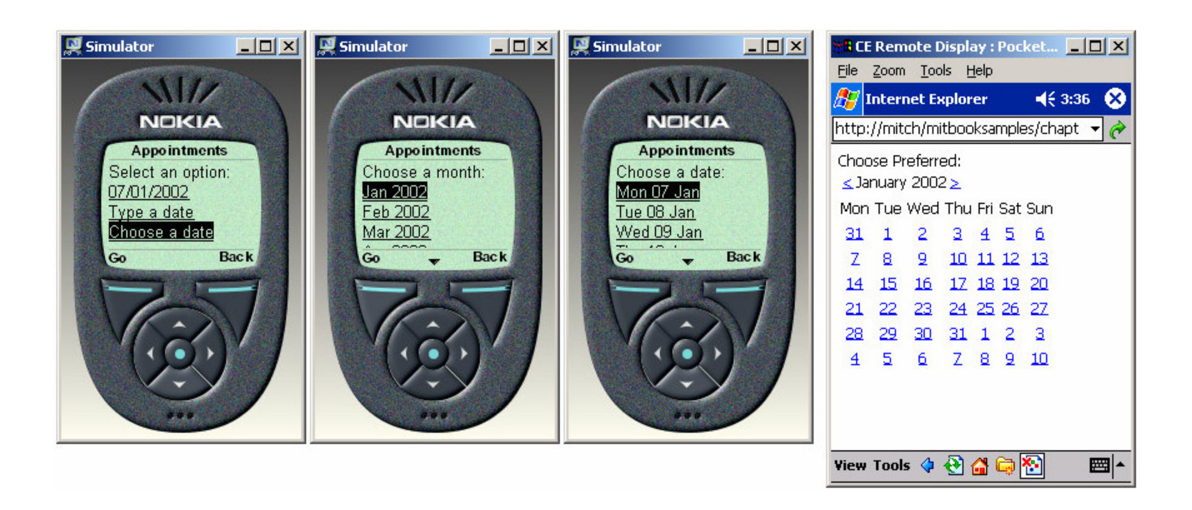
عنصر تحكم Calendar:

يمكن هذا العنصر من مكاملة عملية اختيار التاريخ ضمن تطبيق الوب المحمول حيث تقدم واجهة خاصة من أجل هذه العملية.

يوفر عنصر التحكم هذا عدة وضعيات تحدد ما هوالمجال من التاريخ الذي يمكن للمستخدم الاختيار منه مثلاً يوم، أوأسبوع، أوشهر.

يظهر هذا العنصر غالباً بشكل كامل على التجهيزات التي تدعم HTML وبشكل بنية هرمية من الروابط في حالة التجهيزات المحمولة التي تدعم WML.

فيما يلي لقطة توضح شكل هذا العنصر ضمن مستعرضين مختلفين:



يبين النص البرمجي التالي صيغة استخدام هذا العنصر وأهم الخصائص والأحداث التي يوفرها:

```
<mobile:Calendar
    runat="server"
    id="id"
    BreakAfter="{True|False}"
    Font-Name="fontName"
    Font-Size="{NotSet|Normal|Small|Large}"
    Font-Bold="{NotSet|False|True}"
    Font-Italic="{NotSet|False|True}"
    ForeColor="foregroundColor"
    BackColor="backgroundColor"
    Alignment="{NotSet|Left|Center|Right}"
    StyleReference="styleReference"
    Visible="{True|False}"
    Wrapping="{NotSet|Wrap|NoWrap}"

    CalendarEntryText="prompt string"
    FirstDayOfWeek="{Default|Sunday|Monday|Tuesday|Wednesday|
        Thursday|Friday|Saturday|Sunday}"
    OnSelectionChanged="selectionChangedHandler"
    SelectedDate="selectedDate"
    SelectionMode="{None|Day|DayWeek|DayWeekMonth}"
    ShowDayHeader="{True|False}"
    VisibleDate="visibleDateMonth"
/>
```

نوضح هنا أيضاً الخصائص الإضافية التي لم نقم بتغطيتها أثناء كلامنا عن خصائص عناصر التحكم العامة:

الخاصة	النمط	الوصف
CalendarEntry-Text	String	تقوم بتعيين أوإعادة النص المستخدم في WML و CHTML كرابط للدخول إلى عنصر تحكم Calendar
FirstDayOfWeek	System.Web.UI. WebControls.FirstDay OfWeek enumeration (FirstDayOfWeek.Friday)	تحدد اليوم الأول في الاسبوع الذي ستبدأ عنده قائمة الأيام.
SelectedDate	DateTime	تعين أوتعيد التاريخ الذي تم اختياره بواسطة عنصر التحكم. تكون القيمة الافتراضية هي تاريخ اليوم الحالي.
SelectedDates	غرض SelectedDateCollection	تقوم بإعادة التواريخ المختارة كغرض SelectedDateCollection
SelectionMode	System.Web.UI.WebControls. CalendarSelectionMode None|Day|DayWeek |DayWeekMonth	يحدد وحدات التاريخ التي يستطيع المستخدم اختيارها. إذا تم تحديد هذه القيمة بـ None لن يكون هناك إمكانية للاختيار.
showDayHeader	True | False	قيمة منطقية تحدد إظهار أوإلغاء إضافة اسم اليوم إلى التاريخ المختار.
VisibleDate	DateTimeObject	يقوم بالتحكم بالشهر الذي سيتم إظهاره للمستخدم وذلك بتحديد أي يوم ضمن هذا الشهر.
WebClendar	System.Web.UI.WebControls. Calendar	يغلف الغرض MobileControls.Calendar مثيل من الغرض System.Web.UI.WebControls. Calendar والذي يمكن الوصول إليه من خلال هذه الخاصة.
الحدث SelectionChanged	اسم طريقة معالج الحدث	يحدد اسم طريقة معالج الحدث التي سيتم استدعاؤها عندما يقوم المستخدم بتغيير التاريخ المختار ضمن عنصر التحكم.

يوضح المثال التالي استخدام عنصر التحكم هذا:

```
<%@ Register TagPrefix="mobile"
    Namespace="System.Web.UI.MobileControls"
    Assembly="System.Web.Mobile" %>
<%@ Page language="c#" Codebehind="CalendarExample.aspx.cs"
    Inherits="MSPress.MobWeb.CalEx.CalendarExampleMobileWebForm" %>

<mobile:Form id="Form1" runat="server">
    <mobile:Calendar id="Calendar1" runat="server"
        SelectedDate="2001-07-21"
        SelectionMode="DayWeek"
        Alignment="Center"
        OnSelectionChanged="Calendar1_SelectionChanged">
```

```
    </mobile:Calendar>
    <mobile:Label id="Label1" runat="server" Alignment="Center"/>
</mobile:Form>
```

أما بالنسبة للنص البرمجي للملف CalendarExample.aspx.cs فهو كالتالي:

```
using System;
using System.Web.UI.MobileControls;

namespace MSPress.MobWeb.CalEx
{
    public class CalendarExampleMobileWebForm :
        System.Web.UI.MobileControls.MobilePage
    {
        protected System.Web.UI.MobileControls.Calendar Calendar1;
        protected System.Web.UI.MobileControls.Form Form1;
        protected System.Web.UI.MobileControls.Label Label1;

        override protected void OnInit(EventArgs e)
        {
            InitializeComponent();
            base.OnInit(e);
        }

        private void InitializeComponent()
        {
            this.Calendar1.SelectionChanged +=
                new System.EventHandler(this.
Calendar1_SelectionChanged);
        }

        protected void Calendar1_SelectionChanged(
            object sender,
            System.EventArgs e)
        {
            Label1.Text=Calendar1.SelectedDate.ToShortDateString();
        }
    }
}
```

نلاحظ في المثال استخدام عنصر التحكم Calendar وقد تم إعطاء القيمة الأولية له 21/7/2001 ثم تم تحديد إمكانية الاختيار.

عند اختيار أي تاريخ آخر غير 21/7/2001 سيتم إطلاق الحدث SelectionChanged وبالتالي تشغيل الطريقة calendar1_SelectionChanged التي تقوم بتعيين قيمة عنصر التحكم Label1 إلى التاريخ الجديد.

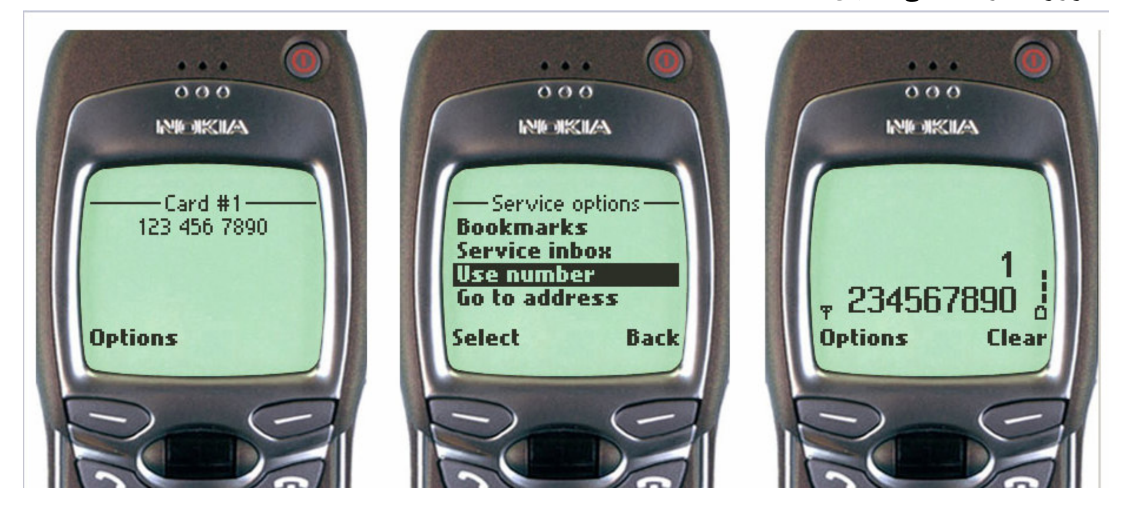
تكون نتيجة تنفيذ هذا النص البرمجي من الشكل:



عناصر تحكم الأغراض الخاصة

عنصر التحكم PhoneCall:

يستخدم عنصر التحكم PhoneCall الطلب المباشر للاتصال الهاتفي في حال كان الهاتف المحمول يدعم هذا الخيار. أما في حال عدم دعم هذه الخاصة يظهر عنصر التحكم هذا وصلة يمكن للمستخدم استخدامها لتفعيل الاتصال أولإظهار رسالة تنبيه المستخدم إلى ضرورة الموافقة على تأسيس هذا الاتصال.



يبين النص البرمجي التالي صيغة استخدام هذا العنصر وأهم الخصائص والأحداث التي يوفرها:

```
<mobile:PhoneCall
    runat="server"
    id="id"
    BreakAfter="{True|False}"
    Font-Name="fontName"
    Font-Size="{NotSet|Normal|Small|Large}"
    Font-Bold="{NotSet|False|True}"
    Font-Italic="{NotSet|False|True}"
    ForeColor="foregroundColor"
    BackColor="backgroundColor"
    Alignment="{NotSet|Left|Center|Right}"
    StyleReference="styleReference"
    Text="text"
    Visible="{True|False}"
    Wrapping="{NotSet|Wrap|NoWrap}"

    AlternateFormat="alternateText"
    AlternateURL="targetURL"
    PhoneNumber="phoneNumber"
    SoftkeyLabel="text"
    Text="text">
innerText
</mobile:PhoneCall>
```

نوضح هنا أيضاً الخصائص الإضافية التي لم نقم بتغطيتها أثناء تكلمنا عن خصائص عناصر التحكم العامة:

الخاصة	النمط	الوصف
AlternateFormat	String	يحدد تنسيق الرسالة التي يجب إظهارها على الجهاز الذي لا يستطيع تأسيس الاتصالات الصوتية. يمكن أن تتضمن سلسلة المحارف المدخلة أحد القيمتين التاليتين {0} و{1}. يتم استبدال خاصة Text بالقيمة {0} والخاصة PhoneNumber. القيمة الافتراضية لهذه الخاصة هي "{0}{1}"
AlternateURL		يعبر عن محدد URL النسبي أوالمطلق للصفحة التي سيجري تحميلها عند عدم تمكن الجهاز من تأسيس الاتصال الهاتفي أوعند عدم رغبة المستخدم بالاتصال.
PhoneNumber	String	يمثل الرقم المراد الاتصال به ممثلاً بالتنسيق رمز البلد|الرمز القطري|الرقم القصير. يمكن أن تتم عملية التنسيق تلك باستخدام الرموز التالية ( ).

		- إضافة إلى الفراغ.
SoftkeyLabel	String	تقدم بعض مستعرضات WML زر برمجي في أسفل الشاشة. يظهر في هذا الزر القيمة الافتراضية Go ويمكننا تغيير هذه القيمة بتحديد SoftkeyLabel إلى القيمة المطلوبة.
Text	String	يحدد الرسالة التي سيتم إظهارها للرابط الخاص بتأسيس الاتصال.

يوضح المثال التالي استخدام عنصر التحكم هذا:

```
<%@ Page Inherits="System.Web.UI.MobileControls.MobilePage"
    Language="c#"%>
<%@ Register TagPrefix="mobile"
    Namespace="System.Web.UI.MobileControls"
    Assembly="System.Web.Mobile" %>

<mobile:Form id="Form1" runat="server">
    <mobile:PhoneCall runat="server"
        AlternateFormat="Call {0} on {1}"
        AlternateURL="http://www.northwindtraders.com"
        phoneNumber="123-456-7890"
        Text="Northwind Traders">
    </mobile:PhoneCall>
</mobile:Form>
```



القسم التاسع والعاشر:

الموضوع الأول:عناصر تحكم نماذج الوب المحمولة

الكلمات المفتاحية:

عنصر تحكم، محمول، لغة تأشير، تأشيرة، واصفة، خاصة، حدث

ملخص:

سنتعرف في هذه الجلسة على عناصر تحكم نماذج الوب المحمولة. ستغطي هذه الجلسة العناصر الحاوية، والأساسية، وتلك المستخدمة للأغراض الخاصة.

أهداف تعليمية:

يتعرف الطالب في هذا الفصل على:

- عناصر تحكم نماذج الوب الحاوية
- عناصر تحكم نماذج الوب الأساسية
- عناصر تحكم نماذج الوب ذات الأغراض الخاصة.

عناصر تحكم الصلاحية

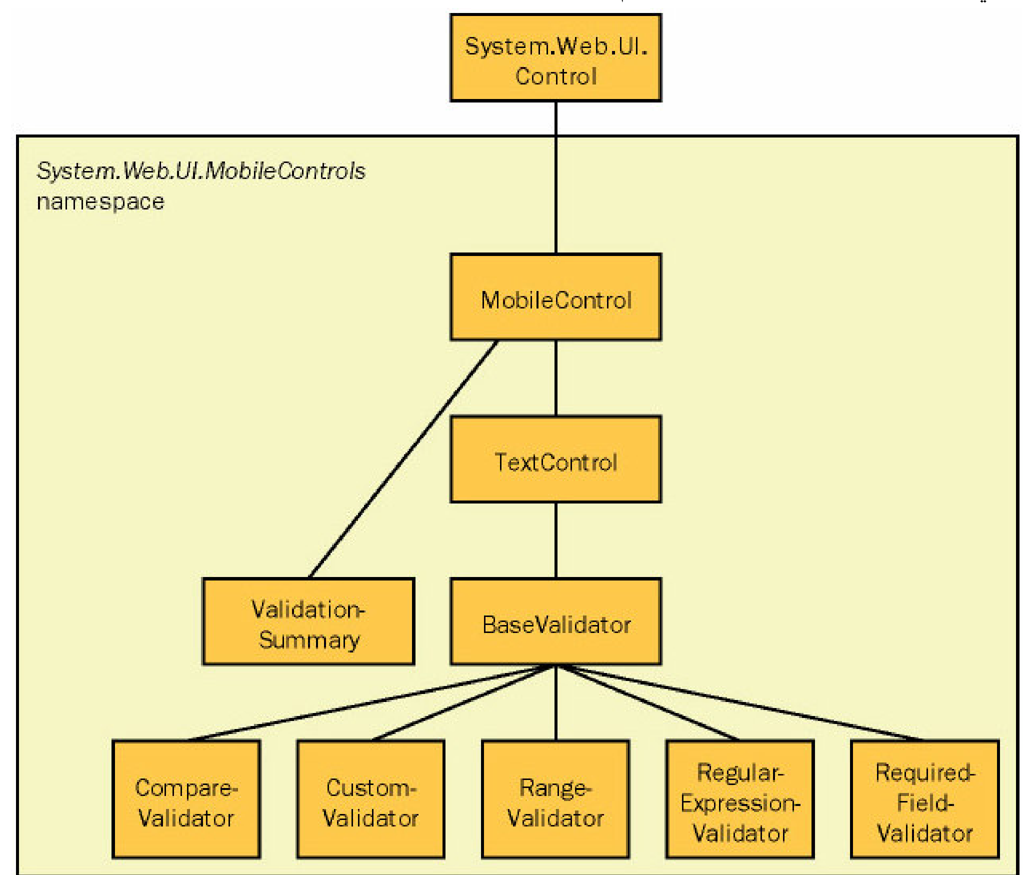
تُعتبر عملية التحقق من صلاحية مدخلات النموذج جزءاً مهم من أي تطبيق. إذ تجري عملية التحقق من الصلاحية لعدة أغراض أهمها :

- التحقق من اكتمال النموذج.
- التحقق من التزام تنسيق معين للبيانات، كحالة البريد الالكتروني مثلاً.
- التحقق من احتواء حقلين على نفس القيمة. كحالة التحقق منحقل كلمة السر ومن حقل تأكيدها.

تقليدياً، يتطلب أداء مثل هذه المهام نصوص برمجية من جهة المخدم. وهي غالباً ما تحتاج إلى استخدام التعابير النظامية المعقدة.

توفر ASP.NET طريقة جديدة لعملية التحقق من صلاحية النماذج باستخدام عناصر تحكم من جهة المخدم تُدعى عناصر تحكم الصلاحية. تسمح هذه العناصر بإجراء عمليات تحقق معقدة بمجرد إدراج التأشيرة الخاصة بعنصر التحكم ضمن صفحة الوب.

يوضح الشكل التالي البنية الهرمية لصفوف عناصر تحكم الصلاحية:



نلاحظ من الشكل السابق وجود 5 عناصر تحكم خاصة بالتحقق من الصلاحية هي:

- عنصر تحكم الحقل الإجباري

- عنصر تحكم التعابير النظامية
- عنصر تحكم صلاحية المجال
- عنصر تحكم الصلاحية المُخصَص
- عنصر تحكم المقارنة.

الخاصات العامة لعناصر تحكم الصلاحية

ترث جميع عناصر تحكم الصلاحية عدا العنصر ValidationSummary، من الصف System.Web.UI.MobileControls.BaseValidator، الخاصةIsValid.

تأخذ هذه الخاصة القيمة True إذا كانت شروط التحقق صحيحة، والقيمة False إذا فشلت عملية التحقق من الصلاحية.

يقدم الصف System.Web.UI.Page (وهوالصف الأب للصف MobilPage) الخاصة IsValid التي تتشكل من عملية (AND) منطقية بين جميع خصائص IsValid لعناصر الصلاحية الموجودة في الصفحة.

تتلخص الطريقة التقليدية في استخدام عناصر تحكم التحقق من الصلاحية باختبار القيمة IsValid للصفحة قبل السماح بالتحرك من تلك الصفحة.

للوصول إلى مرجع عن المثل System.Web.UI.Page من الصف MobilePage نستخدم الخاصة MobilePage.Page.

مثال:

يعرض المثال التالي معالج حدث مرتبط بضغط الزر Command1 والذي يتحقق من قيمة الخاصة IsValid للصفحة قبل الانتقال إلى النموذج Form2:

```
protected void Command1_Click(object sender, System.EventArgs e)
        {
            // Move onto second Form only if input on first page has
            // passed validation by all the validation controls on the
page
            if (Page.IsValid)
            {
                ActiveForm = Form2;
            }
        }
```

في حالة وجود خطأ يظهر النموذج من جديد مع رسالة من أجل كل عنصر تحكم صلاحية تسبب في هذا الخطأ.

تظهر عناصر تحكم الصلاحية الرسائل بشكلين أساسيين:

- الأول في نفس المكان الذي قمت فيه بإدراج عنصر تحكم الصلاحية على النموذج
- والثاني ضمن مساحة العرض الخاصة بعنصر تحكم ValidationSummary.

يجري تحديد رسالة الخطأ التي ستظهر بالشكل الأول باستخدام الخاصة Text. أما في حالة استخدام عنصر التحكم ValidationSummary، فيجري استخدام الخاصة ErrorMessage وفي هذه الحالة يجب إسناد القيمة None إلى قيمة الخاصة Display لعنصر تحكم الصلاحية نفسه، لضمان عدم ظهور رسالة الخطأ مرتين، مرة عن طريق عنصر التحكم ValidationSummary ومرة عن طريق عنصر تحكم الصلاحية نفسه.

يبين الجدول التالي أهم الخصائص المشتركة لعناصر التحقق من الصلاحية:

الخاصة	النمط	الوصف
ControlToValidate	String	مُعرِّف ID لعنصر التحكم المراد التحقق منه.
Display	System.Web.UI.WebControls .ValidatorDisplay None|Static|Dynamic	تحدد إظهار رسالة الخطأ أو لا.
ErrorMessage	String	تعبر عن نص الرسالة التي سيجري إظهارها ضمن خرج عنصر التحكم ValidationSummary
IsValid	True | False	تحدد فيما إذا كانت البيانات صالحة.
Text	String	تمثل الرسالة التي سيجري إظهارها. في حال لم يجر تعيين قيمة لهذه الخاصة، يجري إظهار قيمة الخاصة ErrorMessage عوضاً عنها. لا يتم إدارج هذه القيمة ضمن خرج عنصر التحكم ValidationSummary. لهذا الغرض تم إيجاد القيمة ErrorMessage.

عناصر تحكم الصلاحية

عنصر تحكم الحقل RequiredFieldValidator

يعتبر هذا العنصر أبسط أشكال عناصر تحكم الصلاحية وأكثرها استخداماً. يقوم هذا العنصر ببساطة بالتحقق فيما إذا قام المستخدم بإعطاء قيمة لعنصر إدخال معين أم لا.

تكون الصيغة التي تساعد في استخدام هذا العنصر في صفحة نموذج الوب المحمول هي التالية:

```
<mobile:RequiredFieldValidator
    runat="server"
    id="id"
    BreakAfter="{True|False}"
    Font-Name="fontName"
    Font-Size="{NotSet|Normal|Small|Large}"
    Font-Bold="{NotSet|False|True}"
    Font-Italic="{NotSet|False|True}"
    ForeColor="foregroundColor"
    BackColor="backgroundColor"
    Alignment="{NotSet|Left|Center|Right}"
    StyleReference="styleReference"
    Visible="{True|False}"
    Wrapping="{NotSet|Wrap|NoWrap}"

    ControlToValidate="IdOfTargetControl"
    Display="{None|Static|Dynamic}"
    ErrorMessage="ErrorTextForSummary"
    InitialValue="initialValueInTheControl"
    Text="ErrorText">
innerText
</mobile:RequiredFieldValidator>
```

جميع خصائص هذا العنصر موروثة عن الخصائص العامة لعناصر التحقق من الصلاحية والتي يوفرها الصف BaseValidator.

استخدام عنصر التحكم RequierdFieldValidator:

يشرح المثال التالي استخدام عنصر التحكم RequierdFieldValidator حيث يطلب من المستخدم إدخال اسمه ضمن النموذج وعند إرسال النموذج يجري التحقق من كون الحقل قد تم إعطاؤه قيمة:

```
<%@ Page Inherits="MSPress.MobWeb.ReqEx.RequiredExample"
    CodeBehind="RequiredExample.aspx.cs"
    Language="C#"%>
<%@ Register TagPrefix="mobile"
    Namespace="System.Web.UI.MobileControls"
    Assembly="System.Web.Mobile"  %>

<mobile:Form id="Form1" runat="server">
    <mobile:Label id="Label1" runat="server">
        Your name:
    </mobile:Label>
    <mobile:TextBox id="userName" runat="server"/>
    <mobile:RequiredFieldValidator id="RequiredFieldValidator1"
        runat="server"
        Display="Dynamic"
        ErrorMessage="Your name is required! "
```

```
        ControlToValidate="userName"/>
    <mobile:Command id="Command1" OnClick="Command1_Click"
runat="server">
        Submit
    </mobile:Command>
</mobile:Form>

<mobile:Form id="Form2" runat="server">
    <mobile:Label id="Label2" runat="server">
        Input validated OK.
    </mobile:Label>
</mobile:Form>
```

أما النص البرمجي العامل في الخلفية فهوبالاسم RequiredExample.aspx.cs

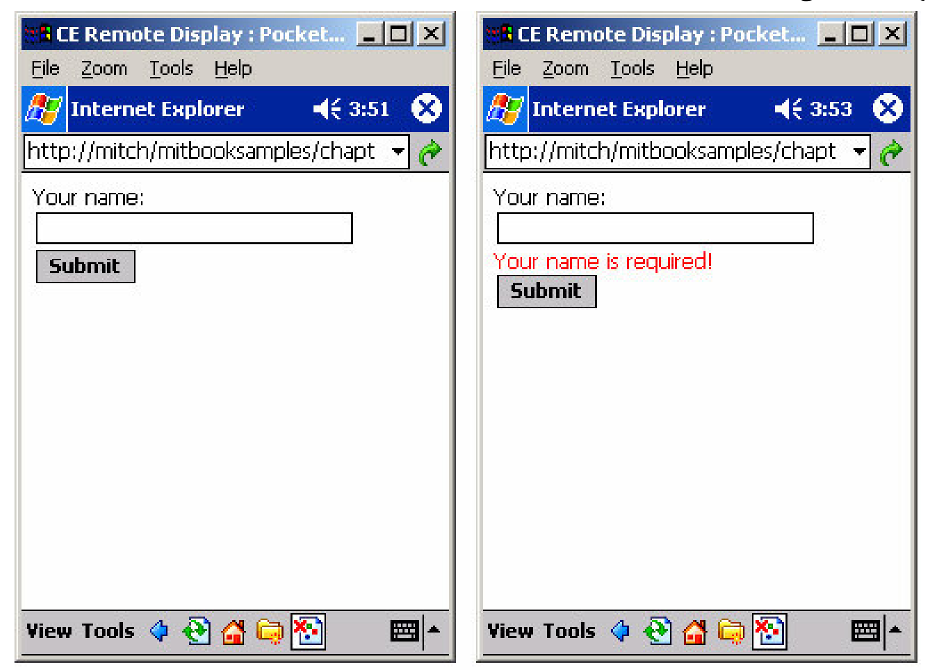
```
using System;

namespace MSPress.MobWeb.ReqEx
{
    public class RequiredExample :
System.Web.UI.MobileControls.MobilePage
    {
        protected System.Web.UI.MobileControls.Label Label1;
        protected System.Web.UI.MobileControls.TextBox userName;
        protected System.Web.UI.MobileControls.RequiredFieldValidator
            RequiredFieldValidator1;
        protected System.Web.UI.MobileControls.Command Command1;
        protected System.Web.UI.MobileControls.Form Form1;
        protected System.Web.UI.MobileControls.Label Label2;
        protected System.Web.UI.MobileControls.Form Form2;
        override protected void OnInit(EventArgs e)
        {
            InitializeComponent();
            base.OnInit(e);
        }

        private void InitializeComponent()
        {
            this.Command1.Click +=
                new System.EventHandler(this. Command1_Click);
        }

        protected void Command1_Click(object sender, System.EventArgs
e)
        {
            if (Page.IsValid)
            {
                ActiveForm = Form2;
            }
        }
    }
}
```

تكون نتيجة التنفيذ لهذا المثال من الشكل



عناصر تحكم الصلاحية

عنصر تحكم المقارنة CompareValidator

يُستخدَم هذا العنصر في عملية مقارنة لقيمتين من قيم عنصر تحكم إدخال.

تكون الصيغة التي تساعد في استخدام هذا العنصر في صفحة نموذج الوب المحمول هي التالية:

```
<mobile:CompareValidator
    runat="server"
    id="id"
    BreakAfter="{True|False}"
    Font-Name="fontName"
    Font-Size="{NotSet|Normal|Small|Large}"
    Font-Bold="{NotSet|False|True}"
    Font-Italic="{NotSet|False|True}"
    ForeColor="foregroundColor"
    BackColor="backgroundColor"
    Alignment="{NotSet|Left|Center|Right}"
    StyleReference="styleReference"
    Visible="{True|False}"
    Wrapping="{NotSet|Wrap|NoWrap}"

    ControlToCompare="IdOfControl"
    ControlToValidate="IdOfTargetControl"
    Display="{None|Static|Dynamic}"
```

```
    ErrorMessage="ErrorTextForSummary"
    Operator="{DataTypeCheck|Equal|GreaterThan|
        GreaterThanEqual|LessThan|
        LessThanEqual|NotEqual}"
    Text="errorText"
    Type="{Currency|DateTime|Double|Integer|String}"
    ValueToCompare="Value">
innerText
</mobile:CompareValidator>
```

يرث عنصر التحكم هذا مجموعة الخصائص العامة التي يوفرها الصف BaseValidator التي مررنا على ذكرها مسبقاً. إضافة إلى هذه الخصائص، يوفر هذا العنصر الخصائص الإضافية التالية:

الخاصة	النمط	الوصف
Operator	System.Web.UI .WebControls .ValidationCompareOperator DataTypeCheck|Equal|GreaterThan |GreaterThanEqual |LessThan|LessThanEqual|NotEqual	تمثل هذه الخاصة عامل المقارنة بين القيمتين المُدخلتين حيث تتوضع القيمة الممثلة لـــ ControlToValidate في الجزء الأيسر لهذا العامل والقيمة الممثلة لـــ ControlToCompare في الجزء الأيمن له
Type	System.Web.UI .WebControls .Validatio-DataType String|Integer|Double|Date|Currency	يقوم بتحديد واستعادة نمط البيانات للقيمتين المراد مقارنتهما. يجري عندها تحويل القيم المدخلة قسرياً إلى نمط البيانات المحدد قبل إجراء عملية المقارنة. أما إذا فشلت عملية التحويل تلك فإن عملية التحقق ستفشل أيضاً.
ValueTo-Compare	String	يجري إسناد قيمة إلى هذه الخاصة إذا أردنا مقارنة الخاصة Text لعنصر تحكم ControlToValidate

مع قيمة ثابتة معينة عوضاً عن مقارنته مع الخاصة Text لعنصر تحكم آخر.		
يمثل المُعرِّف ID لعنصر التحكم المراد المقارنة معه حيث تجري مقارنة قيمتي الخاصتين Text لكل من عنصري التحكم ذوي المحددين ID ControlToCompare و ControlToValidate	String	ControlToCompare

استخدام عنصر التحكم CompareValidator:

يشرح المثال التالي استخدام عنصر التحكم CompareValidator حيث يُطلَب من المستخدم إدخال كلمة السر ضمن عنصر التحكم password1 وإعادة تأكيدها بإدخالها ضمن عنصر تحكم آخر password2. عند إرسال النموذج يجري التحقق من تطابق القيمتين.

```
<%@ Page Inherits="MSPress.MobWeb.CmpEx.CompareExample"
    CodeBehind="CompareExample.aspx.cs"
    Language="C#"%>
<%@ Register TagPrefix="mobile"
    Namespace="System.Web.UI.MobileControls"
    Assembly="System.Web.Mobile" %>

<mobile:Form id="Form1" runat="server">
    <mobile:Label runat="server">
        Your Password
    </mobile:Label>
    <mobile:TextBox id="password1" runat="server" password="true"/>
    <mobile:Label runat="server">
        Retype password
    </mobile:Label>
    <mobile:TextBox id="password2" runat="server" password="true"/>
    <mobile:CompareValidator id="CompareValidator1"
        Type="String"
        Operator="Equal"
        runat="server"
        ErrorMessage="Passwords do not match!"
        ControlToCompare="password1"
        ControlToValidate="password2"/>
    <mobile:Command id="Command1"
        OnClick="Command1_Click" runat="server">
        Submit
```

```
    </mobile:Command>
</mobile:Form>

<mobile:Form id="Form2" runat="server">
    <mobile:Label runat="server">
        Passwords match!
    </mobile:Label>
</mobile:Form>
```

أما النص البرمجي العامل في الخلفية فهوبالاسم CompareExample.aspx.cs

```
using System;

namespace MSPress.MobWeb.CmpEx
{
    public class CompareExample :
System.Web.UI.MobileControls.MobilePage
    {
        protected System.Web.UI.MobileControls.Label Label1;
        protected System.Web.UI.MobileControls.TextBox password1;
        protected System.Web.UI.MobileControls.Label Label2;
        protected System.Web.UI.MobileControls.TextBox password2;
        protected System.Web.UI.MobileControls.CompareValidator
            CompareValidator1;
        protected System.Web.UI.MobileControls.Command Command1;
        protected System.Web.UI.MobileControls.Form Form1;
        protected System.Web.UI.MobileControls.Label Label3;
        protected System.Web.UI.MobileControls.Form Form2;

        override protected void OnInit(EventArgs e)
        {
            InitializeComponent();
            base.OnInit(e);
        }

        private void InitializeComponent()
        {
            this.Command1.Click +=
                new System.EventHandler(this. Command1_Click);
        }
        protected void Command1_Click(object sender, System.EventArgs
e)
        {
            if (Page.IsValid)
            {
                ActiveForm = Form2;
            }
        }
    }
}
```

تكون نتيجة التنفيذ لهذا المثال على الشكل







عناصر تحكم الصلاحية

عنصر تحكم تحديد المجال RangeValidator

يستخدم عنصر التحكم هذا في التحقق من وقوع قيمة ضمن مجال محدد.

تكون الصيغة التي تساعد في استخدام هذا العنصر في صفحة نموذج الوب المحمول هي التالية:

```
<mobile:RangeValidator
    runat="server"
    id="id"
    BreakAfter="{True|False}"
    Font-Name="fontName"
    Font-Size="{NotSet|Normal|Small|Large}"
    Font-Bold="{NotSet|False|True}"
    Font-Italic="{NotSet|False|True}"
    ForeColor="foregroundColor"
    BackColor="backgroundColor"
    Alignment="{NotSet|Left|Center|Right}"
    StyleReference="styleReference"
    Visible="{True|False}"
    Wrapping="{NotSet|Wrap|NoWrap}"

    ControlToValidate="IdOfTargetControl"
    Display="{None|Static|Dynamic}"
    ErrorMessage="ErrorTextForSummary"
    MinimumValue="minValue"
    MaximumValue="maxValue"
    Text="errorText"
    Type="{Currency|DateTime|Double|Integer|String}">
innerText
</mobile:RangeValidator>
```

يرث هذا العنصر مجموعة الخصائص العامة التي يوفرها الصف BaseValidator التي مررنا على ذكرها مسبقاً. إضافة إلى هذه الخصائص يوفر هذا العنصر الخصائص الإضافية التالية:

الخاصة	النمط	الوصف
MinimumValue	String	القيمة الدنيا الواجب توفرها للعنصر ذي المُعرِّف ControlToValidate حتى يجري تقييمه بنجاح.
MaximumValue	String	القيمة الدنيا الواجب توفرها للعنصر ذي المُعرِّف ControlToValidate حتى يجري تقييمه بنجاح. حيث تمثل الخاصتان MinimumValue و MaximumValue قيماً إجبارية يجب إدخالها.
Type	System.Web.UI.WebControls.ValidationDataType String|Integer|Double|Date|Currency	يقوم بتحديد واستعادة نمط البيانات الخاصة بالقيمة المراد مقارنتها. يتم عندها تحويل القيمة المُدخلة قسرياً إلى نمط البيانات المحدد قبل إجراء عملية المقارنة. أما في حال فشل عملية التحويل تلك فإن عملية التحقق ستفشل أيضاً.

استخدام عنصر التحكم RangeValidator:

يشرح المثال التالي استخدام عنصر التحكم RangeValidator الذي يطلب من المستخدم إدخال تاريخ ميلاده ضمن عنصر تحكم TextBox، حيث تكون القيمة العليا المسموح بإدخالها هي تاريخ ميلاد يمتد إلى 21 عاماً من تاريخه أما القيمة الدنيا فهي التاريخ

.1/1/1900

نلاحظ هنا أنه تم تنفيذ الجزء الخاص بإعطاء القيمة البدائية للخاصة MaximumValue ضمن النص البرمجي RangeExample.aspx.cs. فيما يلي النص البرمجي RangeExample.aspx

```
<%@ Page Inherits="MSPress.MobWeb.RgeEx.RangeExample"
    CodeBehind="RangeExample.aspx.cs"
    Language="C#" AutoEventWireup="False" %>
<%@ Register TagPrefix="mobile"
    Namespace="System.Web.UI.MobileControls"
    Assembly="System.Web.Mobile" %>

<mobile:Form id="Form1" runat="server">
    <mobile:Label runat="server">
        Date of birth:
    </mobile:Label>
    <mobile:TextBox id="dob" runat="server"></mobile:TextBox>
    <mobile:RangeValidator id="RangeValidator1" runat="server"
        MinimumValue="01/01/1900"
        ControlToValidate="dob"
        ErrorMessage="Sorry, you are not 21.">
    </mobile:RangeValidator>
    <mobile:Command id="Command1" runat="server" text="Submit">
    </mobile:Command>
</mobile:Form>

<mobile:Form id="Form2" runat="server">
    <mobile:Label id="Label2" runat="server">
        Welcome, you are over 21.
    </mobile:Label>
</mobile:Form>
```

أما النص البرمجي العامل في الخلفية فهوبالاسم RangeExample.aspx.cs

```
using System;

namespace MSPress.MobWeb.RgeEx
{
    public class RangeExample : System.Web.UI.MobileControls.MobilePage
    {
        protected System.Web.UI.MobileControls.RangeValidator
            RangeValidator1;
        protected System.Web.UI.MobileControls.Label Label1;
        protected System.Web.UI.MobileControls.TextBox dob;
        protected System.Web.UI.MobileControls.Command Command1;
        protected System.Web.UI.MobileControls.Form Form1;
        protected System.Web.UI.MobileControls.Label Label2;
        protected System.Web.UI.MobileControls.Form Form2;

        override protected void OnInit(EventArgs e)
        {
```

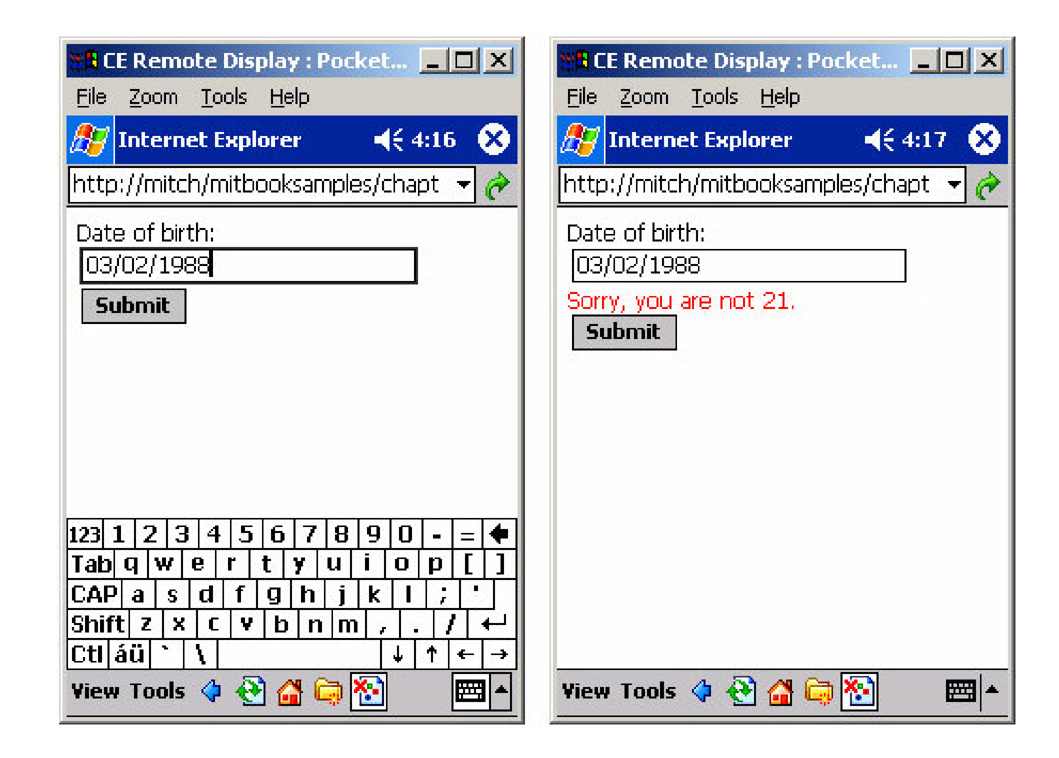
```
            InitializeComponent();
            base.OnInit(e);
        }

        private void InitializeComponent()
        {
            this.Load += new System.EventHandler(this.Page_Load);
            this.Command1.Click +=
                new System.EventHandler(this. Command1_Click);
        }

        protected void Command1_Click(object sender, System.EventArgs
e)
        {
            if (Page.IsValid)
            {
                ActiveForm = Form2;
            }
        }

        private void Page_Load(object sender, System.EventArgs e)
        {
            DateTime now = DateTime.Now;
            DateTime dt21yearsago =
                new DateTime(now.Year - 21, now.Month, now.Day, 0, 0,
0);
            RangeValidator1.MaximumValue =
                dt21yearsago.ToShortDateString();
            RangeValidator1.Type =
                System.Web.UI.WebControls.ValidationDataType.Date;
        }
    }
}
```

تكون نتيجة تنفيذ هذا المثال من الشكل:



عناصر تحكم الصلاحية

عنصر تحكم RegularExpressionValidator

يسمح هذا العنصر باختبار قيمة حقل والتأكد بأنها توافق صيغة معينة. فعلى سبيل المثال، يمكننا استخدام هذا العنصر للتحقق من إدخال قيمة البريد الالكتروني، والرمز البريدي، والرقم الوطني.

يعد هذا العنصر أكثر تعقيداً من عناصر الصلاحية الأخرى التي قمنا بمناقشتها حتى الآن ولكن عمله يظل أقل تعقيداً من محاولة العمل مباشرة مع التعابير النظامية دون استخدام عنصر التحكم هذا.

تكون الصيغة التي تساعد في استخدام عنصر تحكم RegularExpressionValidator في صفحة نموذج الوب المحمول هي التالية:

```
<mobile:RegularExpresssionValidator
    runat="server"
    id="id"
    BreakAfter="{True|False}"
    Font-Name="fontName"
    Font-Size="{NotSet|Normal|Small|Large}"
    Font-Bold="{NotSet|False|True}"
    Font-Italic="{NotSet|False|True}"
```

```
    ForeColor="foregroundColor"
    BackColor="backgroundColor"
    Alignment="{NotSet|Left|Center|Right}"
    StyleReference="styleReference"
    Visible="{True|False}"
    Wrapping="{NotSet|Wrap|NoWrap}"

    ControlToValidate="IdOfTargetControl"
    Display="{None|Static|Dynamic}"
    ErrorMessage="ErrorTextForSummary"
    Text="ErrorText">
    ValidationExpression="regexp" >
innerText
</mobile:RegularExpressionValidator>
```

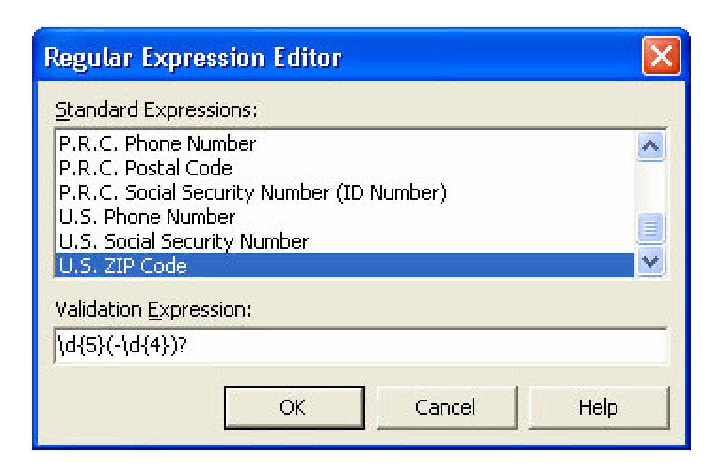
يرث عنصر التحكم هذا مجموعة لخصائص العامة التي يوفرها الصف BaseValidator التي مررنا على ذكرها مسبقاً.

إضافة إلى هذه الخصائص يوفر هذا العنصر الخاصة الإضافية التالية:

الخاصة	النمط	الوصف
ValidationExpression	String	تمثل هذه الخاصة التعبير النظامي الذي يجب تحقيقه.

استخدام عنصر التحكم RegularExpressionValidator:

نظراً لكون عملية كتابة التعبيرات النظامية مربكة بعض الشيء، توفر بيئة Visual Studio معالج يظهر لدى إضافة عنصر RegularExpressionValidator ويوفر مجموعة من أكثر التعبيرات النظامية استخداماً. يظهر هذا المعالج شاشة من الشكل:



يشرح المثال التالي استخدام عنصر التحكم RegularExpressionValidator

فيما يلي النص البرمجي RegularExample.aspx

```
<%@ Page Inherits="MSPress.MobWeb.RegEx.RegularExample"
    CodeBehind="RegularExample.aspx.cs" Language="c#"%>
<%@ Register TagPrefix="mobile"
    Namespace="System.Web.UI.MobileControls"
    Assembly="System.Web.Mobile" %>

<mobile:Form id="Form1" runat="server">
    <mobile:Label runat="server">
        ZIP Code
    </mobile:Label>
    <mobile:TextBox id="zip" runat="server"/>
    <mobile:Command id="Command1" runat="server"
OnClick="Command1_Click">
        Submit
    </mobile:Command>
    <mobile:RegularExpressionValidator
        id="RegularExpressionValidator1"
        runat="server"
        ErrorMessage="Invalid ZIP Code"
        ControlToValidate="zip" ValidationExpression="\d{5}(-\d{4})?"/>
</mobile:Form>

<mobile:Form id="Form2" runat="server">
    <mobile:Label runat="server">
        Valid ZIP Code
    </mobile:Label>
</mobile:Form>
```

أما النص البرمجي العامل في الخلفية فهوبالاسم RangeExample.aspx.cs

```
using System;

namespace MSPress.MobWeb.RegEx
{
    public class RegularExample :
System.Web.UI.MobileControls.MobilePage
    {
        protected System.Web.UI.MobileControls.Label Label1;
        protected System.Web.UI.MobileControls.TextBox zip;
        protected System.Web.UI.MobileControls.Command Command1;
        protected
System.Web.UI.MobileControls.RegularExpressionValidator
                       RegularExpressionValidator1;
        protected System.Web.UI.MobileControls.Form Form1;
        protected System.Web.UI.MobileControls.Label Label2;
        protected System.Web.UI.MobileControls.Form Form2;
```

```
        override protected void OnInit(EventArgs e)
        {
            InitializeComponent();
            base.OnInit(e);
        }

        private void InitializeComponent()
        {
            this.Command1.Click +=
                new System.EventHandler(this. Command1_Click);
        }

        protected void Command1_Click(object sender, System.EventArgs
e)
        {
            if (Page.IsValid)
            {
                ActiveForm = Form2;
            }
        }
    }
}
```

يقوم هذا البرنامج من التأكد من كون قيمة عنصر التحكم TextBox المحدد بقيمة ID=zip يحقق التعبير النظامي "\d{5}(-\d{4})?".

عناصر تحكم الصلاحية

عنصر تحكم CustomValidator

يختلف هذا العنصر عن بقية عناصر التحكم لأنه لايوفر وظيفة التحقق من الصلاحية بصورة مباشرة. ويساعد هذا العنصر في إنشاء طريقة التحقق من الصلاحية.

تكون الصيغة التي تساعد في استخدام عنصر تحكم CustomValidator في صفحة نموذج الوب المحمول هي التالية:

```
<mobile:CustomValidator
    runat="server"
    id="id"
    BreakAfter="{True|False}"
    Font-Name="fontName"
    Font-Size="{NotSet|Normal|Small|Large}"
    Font-Bold="{NotSet|False|True}"
    Font-Italic="{NotSet|False|True}"
    ForeColor="foregroundColor"
    BackColor="backgroundColor"
    Alignment="{NotSet|Left|Center|Right}"
    StyleReference="styleReference"
```

```
    Text="ErrorText"
    Visible="{True|False}"
    Wrapping="{NotSet|Wrap|NoWrap}"

    ControlToValidate="IdOfTargetControl"
    Display="{None|Static|Dynamic}"
    ErrorMessage="ErrorTextForSummary"
    OnServerValidate="EventHandler"
    Text="ErrorText">
innerText
</mobile:CustomValidator>
```

يرث عنصر التحكم هذا مجموعة الخصائص العامة التي يوفرها الصف BaseValidator التي مررنا على ذكرها مسبقاً.

إضافة إلى هذه الخصائص يوفر هذا العنصر حدث إضافي هوالتالي:

الحدث	النمط	الوصف
ServerValidate	طريقة معالج حدث	يجري إطلاق هذا الحدث عند التحقق من الصفحة على المخدم. يستقبل معالج الحدث المعامل ServerValidateEventArgs لذا يجب على معالج الحدث إسناد True إلى قيمة IsValid الخاصة بالغرض ServerValidateEventArgs إذا كانت عملية التحقق صحيحة.

يمكن أن يكون شكل معالج هذا الحدث كما يلي:

```
void ServerValidate (Object source, ServerValidateEventArgs args )
{
    args.IsValid=false;

    // Code to validate the user's input

    if (validationIsSuccessful)
        args.IsValid=true;
}
```

استخدام عنصر التحكم CustomValidator:

يشرح المثال التالي استخدام عنصر التحكم CustomValidator

فيما يلي النص البرمجي CustomExample.aspx

```
<%@ Page Inherits="MSPress.MobWeb.CusEx.CustomExample"
```

```
    CodeBehind="CustomExample.aspx.cs"
    Language="C#"%>
<%@ Register TagPrefix="mobile"
    Namespace="System.Web.UI.MobileControls"
    Assembly="System.Web.Mobile" %>

<mobile:Form id="Form1" runat="server">
    <mobile:Label runat="server">
        Enter an integer
     </mobile:Label>
    <mobile:TextBox id="number" runat="server"/>
    <mobile:CustomValidator id="CustomValidator1"
        runat="server"
        ErrorMessage="Not a factor of four"
        ControlToValidate="number"
        OnServerValidate="ServerValidate"/>
    <mobile:Command id="Command1"
        OnClick="Command1_Click" runat="server">
        Submit
    </mobile:Command>
</mobile:Form>

<mobile:Form id="Form2" runat="server">
    <mobile:Label runat="server">
        A factor of four.
    </mobile:Label>
</mobile:Form>
```

أما النص البرمجي العامل في الخلفية فهوبالاسم RangeExample.aspx.cs

```
using System;
using System.Web.UI.WebControls;

namespace MSPress.MobWeb.CusEx
{
    public class CustomExample :
System.Web.UI.MobileControls.MobilePage
    {
        protected System.Web.UI.MobileControls.Form Form2;
        protected System.Web.UI.MobileControls.Label Label1;
        protected System.Web.UI.MobileControls.CustomValidator
            CustomValidator1;
        protected System.Web.UI.MobileControls.Command Command1;
        protected System.Web.UI.MobileControls.Form Form1;
        protected System.Web.UI.MobileControls.Label Label2;
        protected System.Web.UI.MobileControls.TextBox number;
        override protected void OnInit(EventArgs e)
        {
            InitializeComponent();
```

```
            base.OnInit(e);
        }

        private void InitializeComponent()
        {
            this.Command1.Click +=
                new System.EventHandler(this.Command1_Click);
            this.CustomValidator1.ServerValidate +=
                new ServerValidateEventHandler(this.ServerValidate );
        }

        protected void Command1_Click(object sender, System.EventArgs
e)
        {
            if (Page.IsValid)
            {
                ActiveForm = Form2;
            }
        }

        protected void ServerValidate (
            object source,
            ServerValidateEventArgs args)
        {
            args.IsValid=false;

            try
            {
                int x = Int32.Parse(number.Text);
                if (x % 4==0)
                {
                    args.IsValid=true;
                }
            }
            catch(FormatException e)
            {
                // Exception may be caused by
                // non-integer input on HTML clients
            }
        }
    }
}
```

يقوم هذا البرنامج من التأكد من كون الرقم المدخل في عنصر تحكم TextBox المسمى number هومن مضاعفات العدد 4.

تم كما نرى إضافة عنصر التحكم CustomValidator1 وربط حدث التحقق من الصلاحية على المخدم بالطريقة ServerValidate والتي تم تفصيل عملها ضمن الملف البرمجي CustomExample.aspx.cs

عنصر تحكم ValidationSummary

يقوم هذا العنصر بإعادة ملخص عن الخرج من جميع عناصر التحقق من الصلاحية المستخدمة ضمن نماذج الوب التي تحتويها الصفحة.

يكون لخرج عنصر التحكم هذا فائدة كبيرة في التطبيقات المحمولة كونه يسمح بتشكيل رسالة خطأ ضمن كتلة نصية واحدة مما يسمح بتحسين استخدام التطبيق على التجهيزات ذات مواصفات العرض المحدودة.

تكون الصيغة التي يستخدمها عنصر التحكم هذا هي التالية:

```
<mobile:ValidationSummary
    runat="server"
    id="id"
    BreakAfter="{True|False}"
    Font-Name="fontName"
    Font-Size="{NotSet|Normal|Small|Large}"
    Font-Bold="{NotSet|False|True}"
    Font-Italic="{NotSet|False|True}"
    ForeColor="foregroundColor"
    BackColor="backgroundColor"
    Alignment="{NotSet|Left|Center|Right}"
    StyleReference="styleReference"
    Visible="{True|False}"
    Wrapping="{NotSet|Wrap|NoWrap}"

    BackLabel="BackLabel"
    FormToValidate="FormID"
    HeaderText="HeaderText">
</mobile:ValidationSummary>
```

نلاحظ هنا وجود ثلاث خصائص إضافية غير مألوفة هي :

الخاصة	النمط	الوصف
BackLabel	String	إذا تم إعطاء قيمة لهذه الخاصة فسيجري استخدامها كنص للرابط الذي يعيد المستخدم إلى نموذج الإدخال لإعادة إدخال النص من جديد.
FormToValidate	String	مُعرِّف ID للنموذج الذي يتم التحقق من صلاحيته.
HeaderText	String	تمثل العنوان الذي يسبق قائمة الأخطاء.

		يجري إظهار قيمة هذه الخاصة في أعلى الصفحة في حالة المستعرضات التي تستخدم HTML وقبل كل رسالة خطأ في مستعرضات WML

إذا تم وضع عنصر التحكم ضمن نفس النموذج الذي يحتوي عناصر تحكم التحقق من الصلاحية سيتم بعد إجراء عملية التحقق على المخدم إعادة عنصر التحكم ValidationSummary مظهراً الخاصة ErrorMessage لجميع عناصر التحقق من الصلاحية التي تبين أن قيمة الخاصة IsValid لها هي False.

أما إذا تم وضع عنصر التحكم ValidationSummary ضمن نموذج مغاير لذلك الحاوي على عناصر تحكم التحقق من الصلاحية الأخرى فعندها يجب على معالج حدث الضغط على عنصر تحكم Command الذي قام بإرسال الصفحة اختبار قيمة IsValid للصفحة، فإذا اتضح أن قيمة هذه الخاصة (IsValid) هي False فيجب تعيين قيمة الخاصة ActiveForm بحيث تشير إلى النموذج الحاوي على عنصر التحكم ValidationSummary.في هذه الحالة يجب إعطاء قيمة للخاصة BackLabel بحيث تكون ذات دلالة للمستخدم مثلا Retry.

عند إعطاء قيمة للخاصة BackLabel يقوم العنصر ValidationSummary تلقائياً بالربط مع النموذج الذي يتم التحقق من صلاحيته.

استخدام عنصر التحكم ValidationSummary:

يشرح المثال التالي استخدام عنصر التحكم ValidationSummary

فيما يلي النص البرمجي SummaryExample.aspx

```
<%@ Page Inherits="MSPress.MobWeb.SumEx.SummaryExample"
    CodeBehind="SummaryExample.aspx.cs"
    Language="C#"%>
<%@ Register TagPrefix="mobile"
    Namespace="System.Web.UI.MobileControls"
    Assembly="System.Web.Mobile" %>

<mobile:Form id="Form1" runat="server">
    <mobile:Label id="Label1" runat="server">
        Your name:
    </mobile:Label>
    <mobile:TextBox id="userName" runat="server"/>
    <mobile:Label id="Label2" runat="server" >
        Password
    </mobile:Label>
    <mobile:TextBox id="password" runat="server" Password="True"/>
```

```
    <mobile:RequiredFieldValidator id="RequiredFieldValidator1"
        runat="server"
        ControlToValidate="userName"
        Display="None"
        ErrorMessage="Your name is required!"/>
    <mobile:RequiredFieldValidator id="RequiredFieldValidator2"
        runat="server"
        ControlToValidate="password"
        Display="None"
        ErrorMessage="A password is required!"/>
    <mobile:Command id="Command1" runat="server"
OnClick="Command1_Click">
        Submit
    </mobile:Command>
</mobile:Form>

<mobile:Form id="Form2" runat="server">
    <mobile:ValidationSummary id="ValidationSummary1"
        runat="server"
        HeaderText="Missing Values:"
        FormToValidate="Form1"
        BackLabel="Retry"/>
</mobile:Form>

<mobile:Form id="Form3" runat="server">
    <mobile:Label runat="server">
        Error free submission.
    </mobile:Label>
</mobile:Form>
```

أما النص البرمجي العامل في الخلفية فهوبالاسم SummaryExample.aspx.cs

```
using System;

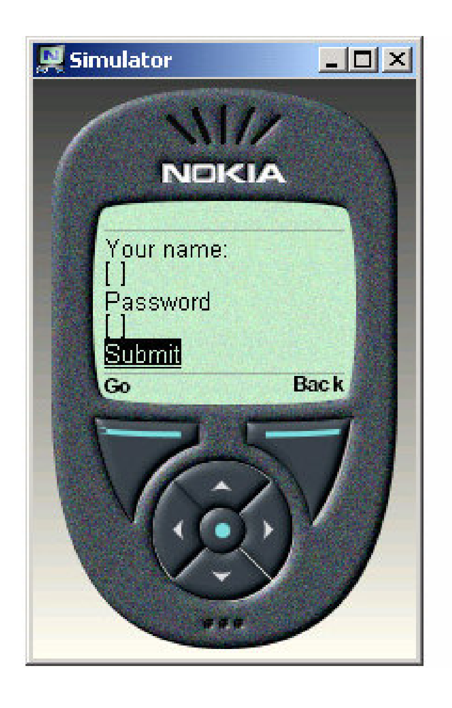
namespace MSPress.MobWeb.SumEx
{
    public class SummaryExample :
System.Web.UI.MobileControls.MobilePage
    {
        protected System.Web.UI.MobileControls.Command Command1;
        protected System.Web.UI.MobileControls.Form Form2;
        protected System.Web.UI.MobileControls.Form Form3;

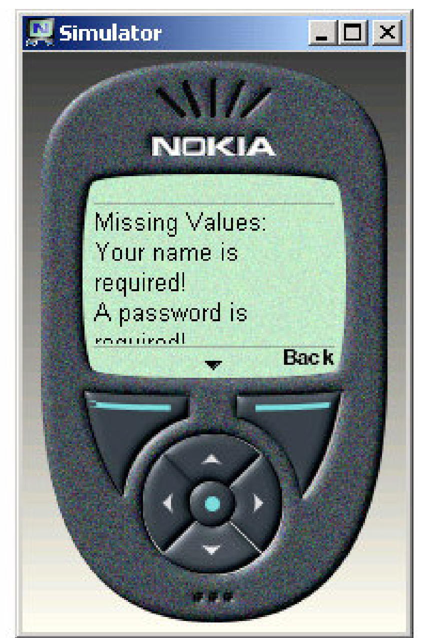
        override protected void OnInit(EventArgs e)
        {
            InitializeComponent();
            base.OnInit(e);
        }
```

```
        private void InitializeComponent()
        {
            this.Command1.Click +=
                new System.EventHandler(this.Command1_Click);
        }

        protected void Command1_Click(object sender, System.EventArgs
e)
        {
            if (Page.IsValid)
            {
                ActiveForm = Form3;
            }
            else
            {
                ActiveForm = Form2;
            }
        }
    }
}
```

أما نتيجة تشغيل هذا النص البرمجي فتكون على الشكل:





مثال شامل حول استخدام عناصر تحكم الصلاحية

سنعمل في هذا المثال على تصميم نموذج إدخال يطلب إدخال معلومات البريد الالكتروني وتأكيد عنوان البريد، ثم تحديد مبلغ للتبرع لمؤسسة خيرية ما . يجب على النص البرمجي التحقق أولاً من إدخال صيغة البريد الالكتروني بالشكل الصحيح، والتطابق بين القيمتين

المدخلتين لهذا البريد. كذلك يجب أن يتم التحقق من كون جميع الحقول لها قيمة قبل عملية الإدخال.

يكون المبلغ الخاص بالتبرع يجب ألا يكون أقل من 5 أكثر من 1000.

شكل الواجهة المتوقع الحصول عليها في هذا المثال:

E-mail address: [ ]
Re-type e-mail [ ]
Donation (min. $5) [ ]
[Donate!]

- E-mail address required
- You must re-type e-mail
- You must enter an amount

الحل:

```
<%@ Page Inherits="MSPress.MobWeb.ValEx.ValidationExample"
    CodeBehind="ValidationExample.aspx.cs" Language="c#" %>
<%@ Register TagPrefix="mobile"
    Namespace="System.Web.UI.MobileControls"
    Assembly="System.Web.Mobile" %>

<mobile:Form id="Form1" runat="server">
    <mobile:Label runat="server" BreakAfter="false">
        E-mail address:
    </mobile:Label>
    <mobile:TextBox id="email1" runat="server"/>
    <mobile:RequiredFieldValidator id="RequiredFieldValidator1"
        runat="server"
        ErrorMessage="E-mail address required"
        ControlToValidate="email1"
        Display="None"/>
    <mobile:RegularExpressionValidator id="RegularExpressionValidator1"
        runat="server"
        ControlToValidate="email1"
        ValidationExpression="\w+([-+.]\w+)*@\w+([-.]\w+)*\.\w+([-
.]\w+)*">
        Not a valid e-mail address
    </mobile:RegularExpressionValidator>

    <mobile:Label id="Label2" runat="server" BreakAfter="false">
        Re-type e-mail
    </mobile:Label>
    <mobile:TextBox id="email2" runat="server"/>
    <mobile:RequiredFieldValidator id="RequiredFieldValidator2"
        runat="server"
        ErrorMessage="You must re-type e-mail"
        ControlToValidate="email2"
```

```
        Display="None"/>
    <mobile:CompareValidator id="CompareValidator1"
        runat="server"
        ErrorMessage="E-mail addresses do not match. "
        ControlToValidate="email2"
        ControlToCompare="email1"
        Display="None"/>

    <mobile:Label id="Label3" runat="server" BreakAfter="false">
        Donation (min. $5)
    </mobile:Label>
    <mobile:TextBox id="donation" runat="server" Password="True"/>
    <mobile:RequiredFieldValidator id="RequiredFieldValidator3"
        runat="server"
        ErrorMessage="You must enter an amount"
        ControlToValidate="donation"
        Display="None"/>
    <!-- The RangeValidator Control requires that a maximum value is
set.
        This value could represent the payment ceiling accepted by the
        online payment service provider -->
    <mobile:RangeValidator id="RangeValidator1"
        runat="server"
        ControlToValidate="donation"
        Type="Currency"
        MinimumValue="5"
        MaximumValue="1000">
    Minimum donation is $5
    </mobile:RangeValidator>

    <mobile:Command id="Command1" runat="server">
        Donate!
    </mobile:Command>
    <mobile:ValidationSummary id="ValidationSummary1"
        runat="server"
        FormToValidate="Form1"/>
</mobile:Form>

<mobile:Form id="Form2" runat="server">
    <mobile:Label id="Label4" runat="server">
        Thank you for donating.
    </mobile:Label>
</mobile:Form>
```

أما النص البرمجي العامل في الخلفية فهو:

```
using System;

namespace MSPress.MobWeb.ValEx
{
    public class ValidationExample :
System.Web.UI.MobileControls.MobilePage
    {
```

```
        protected System.Web.UI.MobileControls.Label Label1;
        protected System.Web.UI.MobileControls.TextBox email1;
        protected System.Web.UI.MobileControls.RequiredFieldValidator
            RequiredFieldValidator1;
        protected
System.Web.UI.MobileControls.RegularExpressionValidator
            RegularExpressionValidator1;
        protected System.Web.UI.MobileControls.Label Label2;
        protected System.Web.UI.MobileControls.TextBox email2;
        protected System.Web.UI.MobileControls.RequiredFieldValidator
            RequiredFieldValidator2;
        protected System.Web.UI.MobileControls.CompareValidator
            CompareValidator1;
        protected System.Web.UI.MobileControls.Label Label3;
        protected System.Web.UI.MobileControls.TextBox donation;
        protected System.Web.UI.MobileControls.RequiredFieldValidator
            RequiredFieldValidator3;
        protected System.Web.UI.MobileControls.RangeValidator
            RangeValidator1;
        protected System.Web.UI.MobileControls.Command Command1;
        protected System.Web.UI.MobileControls.ValidationSummary
            ValidationSummary1;
        protected System.Web.UI.MobileControls.Form Form1;
        protected System.Web.UI.MobileControls.Label Label4;
        protected System.Web.UI.MobileControls.Form Form2;

        override protected void OnInit(EventArgs e)
        {
            InitializeComponent();
            base.OnInit(e);
        }

        private void InitializeComponent()
        {
            this.Command1.Click +=
                new System.EventHandler(this.Command1_Click);
        }

        protected void Command1_Click(object sender, System.EventArgs
e)
        {
            if (Page.IsValid)
            {
                ActiveForm = Form2;
            }
        }
    }
}
```

القسم التاسع والعاشر:

الموضوع الثاني: عناصر تحكم نماذج الوب الخاصة بالقوائم

الكلمات المفتاحية:

عنصر تحكم، محمول، لغة تأشير، تأشيرة، واصفة، خاصة، حدث

ملخص:

سنتعرف في هذه الجلسة على عناصر تحكم نماذج الوب الخاصة بالقوائم، حيث ستغطي هذه الجلسة عناصر تحكم القائمة SelectionList , List , ObjectList.

أهداف تعليمية:

يتعرف الطالب في هذا الفصل على:

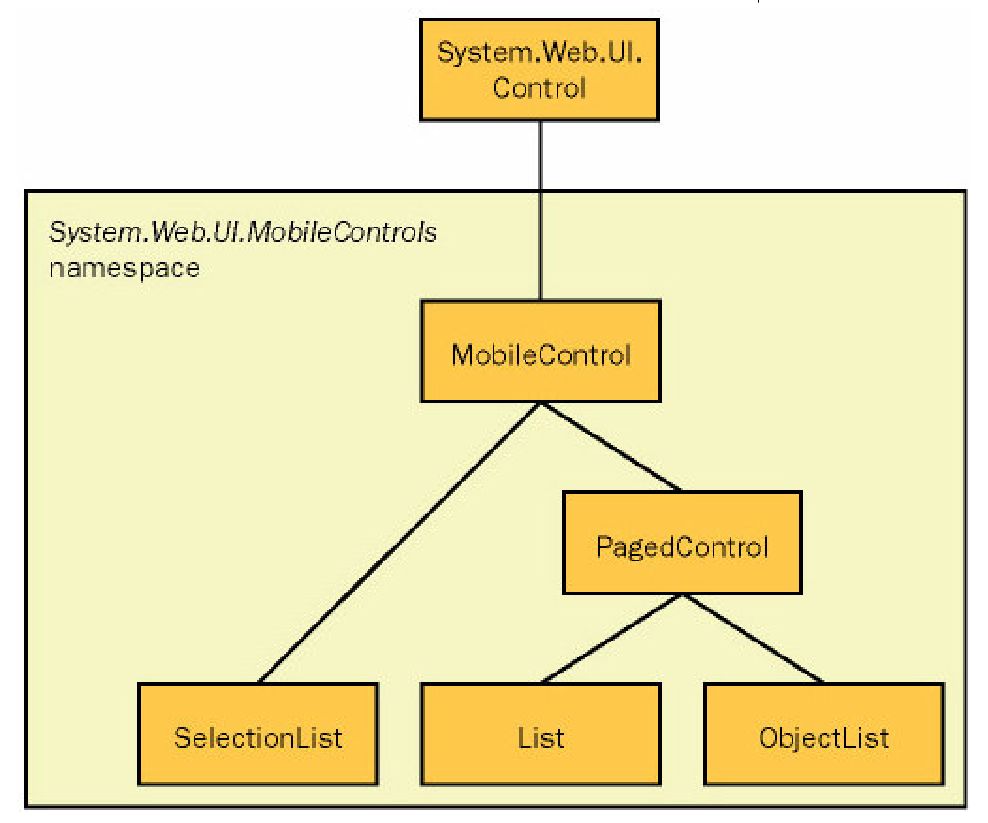
- عنصر التحكم SelectionList
- عنصر التحكم List
- عنصر التحكم ObjectList.

عناصر تحكم القائمة

سنغطي في هذا الجزء وظائف ثلاثة عناصر تحكم تقدم للمستخدم قائمة من الأغراض:

- عنصر تحكم SelectionList
- عنصر تحكم ListControl
- عنصر تحكم ObjectList.
-

وللوصول لفهم أكبر حول هذه عناصر نبين هرمية الصفوف الخاصة بتلك العناصر:



فيما يلي قائمة تشمل أهم الفروقات والمزايا التي يقدمها كل عنصر من عناصر التحكم السابقة:

ObjectList	List	SelectionList	الميزة
X	X	√	يمكن إظهاره بشكل قائمة منسدلة أوقائمة اختيار أوخيارات راديوعلى مستعرضات HTML
		√	يدعم تعدد الخيارات
√	√		يمكن إظهاره على شكل قائمة ثابتة غير تفاعلية
	√		يمكن إظهاره على شكل قائمة تعداد نقطي أورقمي

يدعم عملية تقسيم القوائم الطويلة إلى صفحات		√	√
يمكن التصريح عن عناصر الإظهار بصورة ساكنة		√	√
يمكن ربطها بمصدر بيانات	√	√	√
يمكنها إظهار حقلين أو أكثر من عنصر بيانات			√
تقوم بإطلاق حدث لدى اختيار عنصر	√	√	√
تدعم الإظهار المخصص باستخدام القوالب		√	√

عناصر تحكم القائمة

عنصر تحكم SelectionList

يناسب عنصر تحكم SelectionList القوائم الصغيرة كونه لا يدعم عملية التقطيع إلى صفحات بالنسبة للقوائم الطويلة. ولكن يمكن لهذا العنصر إظهار القائمة بشكل قائمة منسدلة أوقائمة عناصر راديوعلى المستعرضات التي تدعم هذه العناصر.

يعرض عنصر التحكم قائمة مكونة من عمود واحد كما يدم إمكانية ربط قيمة مخفية بكل قيمة ظاهرة. يمكننا تحديد هذه القيمة من خلال الواصفة Value المرتبطة بالعنصر <item> أوتحديد قيمة الخاصة DataValueField في حالة القائمة المرتبطة بمصدر بيانات.

تكون صيغة استخدام عنصر التحكم هذا على الشكل:

```
<mobile:SelectionList
   runat="server"
   id="id"
   Alignment="{NotSet|Left|Center|Right}"
   BackColor="backgroundColor"
   BreakAfter="{True|False}"
   Font-Bold="{NotSet|False|True}"
   Font-Italic="{NotSet|False|True}"
   Font-Name="fontName"
   Font-Size="{NotSet|Normal|Small|Large}"
   ForeColor="foregroundColor"
   StyleReference="StyleReference"
   Wrapping="{NotSet|Wrap|NoWrap}"

   DataMember="dataMember"
   DataSource="dataSource"
   DataTextField="DataTextField"
   DataValueField="DataValueField"
   SelectType="{DropDown|ListBox|Radio|MultiSelectListBox|CheckBox}"
   Title="String"
   OnItemDataBind="itemDataBindHandler"
   OnSelectedIndexChanged="selectedIndexChangedHandler">

       <!-- Optional statically declared list items -->
       <Item Text="Text" Value="Value" Selected="{True|False}"/>

</mobile:SelectionList>
```

الخصائص والأحداث:

فيما يلي أهم الخصائص والأحداث التي يقدمها عنصر التحكم SelectionControl

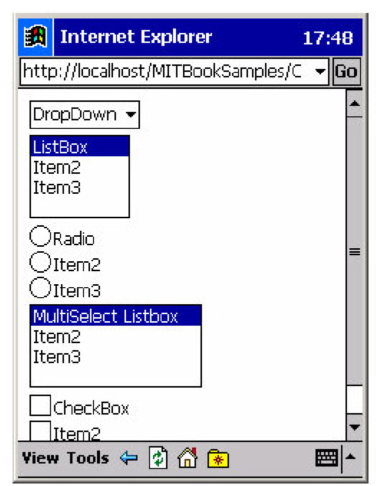
الخاصة أو الحدث	النمط	الوصف
DataMember	String	تُستخدم هذه الخاصة عندما يكون عنصر التحكم مرتبطاً بغرض من نوع System.Data.DataSet أو System.Data.DataTableObject
DataSource	Object	تحدد هذه الخاصة الغرض DataSet أو Collection الذي سيتم الربط معه عندما يكون عنصر التحكم مرتبطاً بمصدر بيانات.
DataTextField	String	في حال ربط عنصر التحكم بغرض DataSet أو بغرض Collection تحدد الخاصة DataTextField اسم الحقل الذي ستظهر محتوياته ضمن القائمة ضمن مصدر البيانات.
DataValueField	String	في حال ربط عنصر التحكم بغرض DataSet أو بغرض Collection تحدد الخاصة DataValueField اسم الحقل الذي ستُستخدم محتوياته ضمن قيم البيانات المخفية المرتبطة مع كل عنصر ضمن القائمة
Items	System.Web.UI.MobileControls .MobileListItemCollection	توفر هذه الخاصة الوصول إلى غرض MobileListItemCollection الذي يحتوي جميع أغراض System.Web.UI.MobileControls. MobileListItem المعبرة عن أغراض القائمة المخزنة.
Rows	Integer	عندما يتم إسناد ListBox أو MultiSelectListBox إلى الخاصة SelectType، يجري استخدام الخاصة Rows لتحديد عدد الصفوف المرئية من عنصر التحكم وذلك في المستعرضات

		التي تدعم HTML و CHTML
SelectedIndex	Integer	تقوم بإعادة أوتحديد دليل العنصر المختار. أما في حال كان عنصر التحكم في وضعية الاختيار المتعدد فيجري إعادة دليل العنصر الأول المختار.
Selection	MobileListItem	تقوم بإعادة غرض العنصر المختار وهومن النمط MobileListItem أوتعيد null في حال لم يتم اختيار أي عنصر.
SelectType	System.Web.UI. MobileControls. ListSelectType DropDown|ListBox|Radio| MultiSelectList-Box|CheckBox	تحدد هذه الخاصة نمط إظهار القائمة في حال دعم المستعرض للنمط المختار.
Title	String	العنوان الذي يظهر على مستعرضات WML. هذا العنوان غير مدعوم في العديد من مستعرضات WML
الحدث ItemDataBind	طريقة معالج حدث	تحدد اسم طريقة معالج الحدث OnItemDataBind(Object Sender , ListDataBindEventsArgs e) يجري إطلاق هذا الحدث عندما يجري ربط عنصر التحكم مع مصدر بيانات. تجري عملية إعادة إطلاق هذا الحدث عند إضافة عنصر إلى القائمة.
الحدث SelectedIndexChanged	طريقة معالج حدث	إذا تم إعداد عنصر التحكم SelectionList للعمل بوضعية الاختيار الوحيد، يجري إطلاق هذا الحدث في كل مرة تجري فيها عملية إرسال النموذج مع تغيير في العنصر المختار.

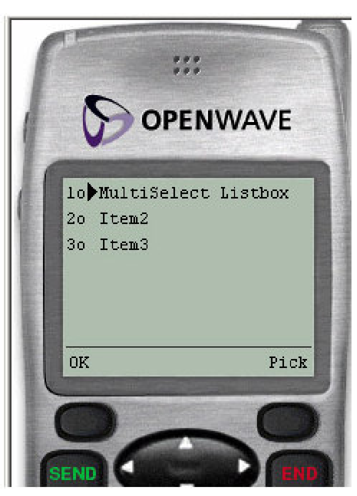
استخدام عنصر التحكم SelectionControl:

لاستخدام هذا العنصر يمكننا تحديد عناصر القائمة باستخدام التأشيرة <item> ضمن التأشيرة الخاصة بهذا العنصر أو بربط العنصر بمصدر بيانات.

يمكننا تحديد نمط عنصر التحكم هذا من خلال الخاصة SelectType حيث تظهر الأنماط المختلفة تبعاً دعم المستعرض للنمط.







مثال1:

يبين المثال التالي استخدام عنصر التحكم:

```
<%@ Page Inherits="MSPress.MobWeb.SelListEx.ExampleWebForm"
Language="c#"
    CodeBehind=" SingleSelectionListExample.aspx.cs"%>
<%@ Register TagPrefix="mobile"
    Namespace="System.Web.UI.MobileControls"
    Assembly="System.Web.Mobile" %>

<mobile:Form runat="server" id="Form1">
   <mobile:Label runat="server" StyleReference="title" id="Label1">
       Season 2003 results
   </mobile:Label>
   <mobile:Label runat="server" id="Label2">Select a
team:</mobile:Label>
   <mobile:SelectionList SelectType="ListBox"
       id="SelectionList1" runat="server">
       <item Text="Dunes" Value="Posn:1 Pl:38 Pts:80"/>
       <item Text="Phoenix" Value="Posn:2 Pl:38 Pts:70"/>
       <item Text="Eagles" Value="Posn:3 Pl:38 Pts:69"/>
       <item Text="Zodiac" Value="Posn:4 Pl:38 Pts:68"/>
   </mobile:SelectionList>
   <mobile:Command runat="server" id="Command1">
       Get Stats!
    </mobile:Command>
</mobile:Form>

<mobile:Form runat="server" id="Form2">
   <mobile:Label runat="server" id="Label3">Team Full
Stats:</mobile:Label>
   <mobile:Label runat="server" id="Label4"/>
</mobile:Form>
```

يقوم النص البرمجي السابق بإنشاء نموذجين: يحتوي الأول قائمة تستخدم SelectionList تظهر معلومات إضافية عنها ضمن الواصفة Value المرتبطة بالتأشيرة <Item>.

فيما يلي النص العامل في الخلفية SingleSelectionListExample.aspx.cs

```
using System;
namespace MSPress.MobWeb.SelListEx
{
    public class ExampleWebForm :
System.Web.UI.MobileControls.MobilePage
    {
        protected System.Web.UI.MobileControls.Label Label4;
        protected System.Web.UI.MobileControls.SelectionList
SelectionList1;
        protected System.Web.UI.MobileControls.Command Command1;
        protected System.Web.UI.MobileControls.Form Form2;

        override protected void OnInit(EventArgs e)
        {
            InitializeComponent();
            base.OnInit(e);
        }

        private void InitializeComponent()
        {
            this.Command1.Click +=
                new System.EventHandler(this.HandleTeamSelection);
        }

        private void HandleTeamSelection(Object source, EventArgs args)
        {
            // Display the Stats page
            this.ActiveForm = Form2;
            String selectedTeamStats = SelectionList1.Selection.Value;
            Label4.Text = SelectionList1.Selection + ": "
+ selectedTeamStats;
        }
    }
}
```

مثال 2:

لوحاولنا إعادة هذا المثال لتقديم إمكانية الاختيار المتعدد، يصبح مثالنا على الشكل:

```
<%@ Page Inherits="MSPress.MobWeb.MultSelListEx.ExampleMobileWebForm"
    Language="c#" CodeBehind="multipleselectionlistexample.aspx.cs"%>
<%@ Register TagPrefix="mobile"
    Namespace="System.Web.UI.MobileControls"
```

```
    Assembly="System.Web.Mobile" %>

<mobile:Form runat="server" id="Form1">
    <mobile:Label runat="server" StyleReference="title">
        Season 2003 results
    </mobile:Label>
    <mobile:Label runat="server">Select 2 or more teams:</mobile:Label>
    <mobile:SelectionList SelectType="MultiSelectListBox"
                          id="SelectionList1" runat="server">
        <item Text="Dunes" Value="Posn:1 Pl:38 Pts:80"/>
        <item Text="Phoenix" Value="Posn:2 Pl:38 Pts:70"/>
        <item Text="Eagles" Value="Posn:3 Pl:38 Pts:69"/>
        <item Text="Zodiac" Value="Posn:4 Pl:38 Pts:68"/>
    </mobile:SelectionList>
    <mobile:Command runat="server" id="Command1">
        Compare Stats!
    </mobile:Command>
</mobile:Form>

<mobile:Form runat="server" id="Form2">
    <mobile:Label runat="server">Teams Full Stats:</mobile:Label>
    <mobile:TextView runat="server" id="TextView1"/>
</mobile:Form>
```

أما النص البرمجي العامل في الخلفية فيكون:

```
using System;
using System.Web.UI.MobileControls;

namespace MSPress.MobWeb.MultSelListEx
{
    public class ExampleMobileWebForm :
        System.Web.UI.MobileControls.MobilePage
    {
        protected System.Web.UI.MobileControls.TextView TextView1;
        protected System.Web.UI.MobileControls.SelectionList
SelectionList1;
        protected System.Web.UI.MobileControls.Command Command1;
        protected System.Web.UI.MobileControls.Form Form2;
        override protected void OnInit(EventArgs e)
        {
            InitializeComponent();
            base.OnInit(e);
        }

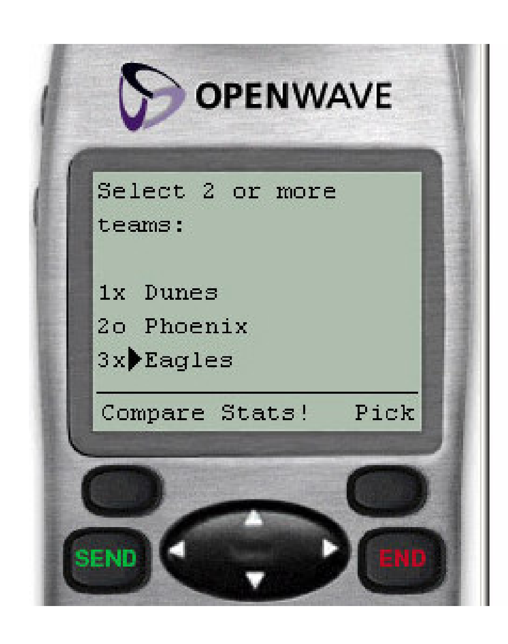
        private void InitializeComponent()
        {
            this.Command1.Click +=
                new System.EventHandler(this.HandleMultiTeamSelection);
        }

        protected void HandleMultiTeamSelection(Object source,EventArgs
```

```
args)
        {
            this.ActiveForm = Form2;
            // Get the list items collection.
            MobileListItemCollection colItems = SelectionList1.Items;
            String strDisplaytext = "";
            foreach (MobileListItem item in colItems)
            {
                if (item.Selected)
                {
                  strDisplaytext += (item.Text + ": " + item.Value +
"<BR>");
                }
            }
            TextView1.Text = strDisplaytext;
        }
    }
}
```

حيث قمنا باستخدام التركيب foreach للتحقق من العناصر المختارة وإظهارها.

تظهر نتيجة تنفيذ المثال بعد التعديل كما في الشكل التالي:





مثال 2:

يتناول المثال التالي نفس الفكرة السابقة مع عملية ربط لعنصر التحكم SelectionList مع مصدر بيانات والذي يمثله هنا الغرض array من النمط ArrayList:

```
<%@ Page Inherits="MSPress.MobWeb.DBListEx.ExampleWebForm"
Language="c#"
    CodeBehind="DataboundListExample.aspx.cs" AutoEventWireup="False"
%>
<%@ Register TagPrefix="mobile"
    Namespace="System.Web.UI.MobileControls"
    Assembly="System.Web.Mobile" %>

<mobile:Form runat="server" id="Form1">
    <mobile:Label id="Label1" runat="server" StyleReference="title">
        Season 2003 results
    </mobile:Label>
    <mobile:Label id="Label2" runat="server">
        Select 2 or more teams:
    </mobile:Label>
    <mobile:SelectionList id="SelectionList1" runat="server"
        DataValueField="Stats" DataTextField="TeamName"
        SelectType="MultiSelectListBox">
    </mobile:SelectionList>
    <mobile:Command id="Command1" runat="server">
        Compare Stats!
    </mobile:Command>
</mobile:Form>

<mobile:Form runat="server" id="Form2">
    <mobile:Label id="Label3" runat="server">Teams Full
Stats:</mobile:Label>
    <mobile:TextView id="TextView1" runat="server"></mobile:TextView>
</mobile:Form>
```

أما النص البرمجي العامل في الخلفية فهو:

```
using System;
using System.Collections;
using System.Web.UI.MobileControls;

namespace MSPress.MobWeb.DBListEx
{
    public class ExampleWebForm :
System.Web.UI.MobileControls.MobilePage
    {
        protected System.Web.UI.MobileControls.TextView TextView1;
        protected System.Web.UI.MobileControls.SelectionList
SelectionList1;
        protected System.Web.UI.MobileControls.Command Command1;
        protected System.Web.UI.MobileControls.Form Form2;
```

```
        override protected void OnInit(EventArgs e)
        {
            InitializeComponent();
            base.OnInit(e);
        }

        private void InitializeComponent()
        {
            this.Load += new System.EventHandler(this.Page_Load);
            this.Command1.Click +=
                new System.EventHandler(this.HandleMultiTeamSelection);
        }
        private void Page_Load(Object sender, EventArgs e)
        {
            if (!IsPostBack)
            {
                ArrayList array = new ArrayList();
                array.Add(new TeamStats("Dunes", "Posn:1 Pl:38
Pts:80"));
                array.Add(new TeamStats("Phoenix", "Posn:2 Pl:38
Pts:70"));
                array.Add(new TeamStats("Eagles", "Posn:3 Pl:38
Pts:69"));
                array.Add(new TeamStats("Zodiac", "Posn:4 Pl:38
Pts:68"));
                SelectionList1.DataSource = array;
                SelectionList1.DataBind();
            }
        }
        private void HandleMultiTeamSelection(
            Object source, EventArgs args)
        {
            this.ActiveForm = Form2;

            // Get the list items collection.
            MobileListItemCollection colItems = SelectionList1.Items;
            String strDisplaytext = "";
            foreach (MobileListItem item in colItems)
            {
                if (item.Selected)
                {
                  strDisplaytext += (item.Text + ": " + item.Value +
                  "<br/>");
                }
            }
            TextView1.Text= strDisplaytext;
        }
    }

    class TeamStats
    {
```

```
        private String teamName, stats;

        public TeamStats(String teamName, String stats)
        {
            this.teamName = teamName;
            this.stats = stats;
        }

        public String TeamName { get { return this.teamName; } }
        public String Stats    { get { return this.stats; } }
    }
}
```

عناصر تحكم القائمة

عنصر تحكم List

يشبه هذا العنصر عنصر تحكم SelectionList الذي قمنا بمناقشته سابقاً، بفرق أن عنصر التحكم List يدعم عملية تقسيم خرج القائمة إلى عدة صفحات مما يقدم فائدة كبيرة في حالة القوائم الطويلة على الأجهزة المحمولة ذات شاشات الإظهار المحدودة الإمكانيات.

يمكننا استخدام هذا العنصر أيضاً في حالة الرغبة بعرض قوائم تفاعلية وغير تفاعلية. في حالة القوائم التفاعلية لسنا بحاجة إلى زر خاص لعملية إرسال البينات إلى المخدم، إذ يتولى عنصر التحكم هذا عملية إرسال البيانات لدى النقر واختيار أحد العناصر.

يدعم عنصر التحكم List القوالب مما يجعل عنصر التحكم هذا مناسباً في حالة برمجة الاستجابات المتعددة المرتبطة بنوع الجهاز المستخدم.

تكون الصيغة الخاصة باستخدام عنصر التحكم هذا هي التالية:

```
<mobile:List
   runat="server"
   id="id"
   Alignment="{NotSet|Left|Center|Right}"
   BackColor="backgroundColor"
   Font-Bold="{NotSet|False|True}"
   Font-Italic="{NotSet|False|True}"
   Font-Name="fontName"
   Font-Size="{NotSet|Normal|Small|Large}"
   ForeColor="foregroundColor"
   StyleReference="StyleReference"
   Wrapping="{NotSet|Wrap|NoWrap}"

   DataMember="dataMember"
   DataSource="dataSource"
   DataTextField="DataTextField"
   DataValueField="DataValueField"
   Decoration="{None|Bulleted|Numbered}"
```

```
    ItemsAsLinks="{False|True}"
    ItemCount="itemCount"
    OnItemDataBind="onItemDataBindHandler"
    OnItemCommand="onItemCommandHandler"
    OnLoadItems="loadItemsHandler">

    <!-- Optional statically declared list items -->
    <Item Text="Text" Value="Value" Selected="{True|False}" />

</mobile:List>
```

يقدم عنصر التحكم هذا كما نرى مجموعة من الخصائص والأحداث نذكر أهمها:

DataMember	String	تُستخدم هذه الخاصة عند ارتباط عنصر التحكم هذا بمصدر بيانات مثل غرض DataSet أو DataTable. حيث تحدد اسم الجدول ضمن غرض DataSet الذي يجب ربط عنصر التحكم به.
DataSource	Object	عند ارتباط هذا العنصر بمصدر بيانات، تحدد هذه الخاصة غرض Collection أو DataSet يمثل مصدر البيانات.
DataTextField	String	عندما يكون عنصر التحكم مرتبطاً بغرض DataSet أو Collection تحدد هذه الخاصة اسم الحقل ضمن مصدر البيانات الذي سيتم إظهاره ضمن القائمة.
DataValueField	String	عندما يكون عنصر التحكم مرتبطاً بغرض DataSet أو Collection تحدد هذه الخاصة اسم الحقل ضمن مصدر البيانات الذي سيقدم القيم المخفية المرتبطة بكل عنصر من عناصر القائمة.
Decoration	System.Web.UI .MobileControls. ListDecoration None|Bulleted|Numbered	تحدد هذه الخاصة على المستعرضات التي تدعم HTML شكل إظهار عناصر القائمة.
ItemsAsLinks	False | True	تستخدم في حالات خاصة عندما نريد استخدام قيم Text و Value لكل عنصر في القائمة بمثابة نص ارتباط ومحدد URL الخاص به. عند تكون قيمة هذه الخاصة True يقوم المستعرض بالانتقال مباشرةً إلى المصدر المحدد ولا يجر توليد أي حدث اختيار
ItemCount	Integer	تعيد هذه الخاصة العدد الكامل لعناصر القائمة.

الحدث ItemCommand	اسم طريقة معالج الحدث	يحدد معالج الحدث الواجب استدعاؤه عندما يختار المستخدم عنصر من القائمة إلا في حال تعيين قيمة الخاصة ItemsAsLinks إلى True
الحدث LoadItems	اسم طريقة معالج الحدث	يتم إطلاق هذا الحدث عند محاولة تحميل عناصر جديدة إلى القائمة من مصدر البيانات. تتم هذه العملية عادة عند تفعيل إمكانية التقسيم إلى صفحات حيث تترافق كل عملية تحميل بانطلاق لهذا الحدث.

استخدام عنصر التحكم List:

ذكرنا سابقاً أنه يمكن استخدام هذا العنصر بوضعيتين:

- الوضعية غير التفاعلية: يتم في هذه الوضعية إظهارعناصر القائمة دون القدرة على الاختيار. يمكن هنا تحديد شكل الإظهار باستخدام الخاصة Decoration. ويجب عدم تحديد قيمة لمعالج الحدث ItemCommand في هذه الحالة.
- الوضعية التفاعلية: في حال تم تعيين قيمة لمعالج الحدث ItemCommand يجري تحويل كل عنصر من عناصر القائمة لرابط يقوم باستدعاء معالج الحدث عند النقر عليه.

مثال1:

يوضح المثال التالي استخدام عنصر التحكم هذا:

```
<%@ Page Inherits="MSPress.MobWeb.ListItmCmd.MyWebForm" Language="c#"
    CodeBehind="ListItemCommandExample.aspx.cs"%>
<%@ Register TagPrefix="mobile"
    Namespace="System.Web.UI.MobileControls"
    Assembly="System.Web.Mobile" %>

<mobile:Form runat="server" id="Form1">
    <mobile:Label runat="server" id="Label1" StyleReference="title">
        Season 2003 results
    </mobile:Label>
    <mobile:Label runat="server" id="Label2">Select a
team:</mobile:Label>
    <mobile:List runat="server" id="List1">
        <item Text="Dunes" Value="Posn:1 Pl:38 Pts:80"/>
        <item Text="Phoenix" Value="Posn:2 Pl:38 Pts:70"/>
        <item Text="Eagles" Value="Posn:3 Pl:38 Pts:69"/>
        <item Text="Zodiac" Value="Posn:4 Pl:38 Pts:68"/>
    </mobile:List>
</mobile:Form>

<mobile:Form runat="server" id="Form2">
    <mobile:Label runat="server" id="Label3" StyleReference="title">
        Team Full Stats:
    </mobile:Label>
    <mobile:Label runat="server" id="Label4" />
</mobile:Form>
```

أما النص البرمجي العامل في الخلفية فهو:

```
using System;
using System.Web.UI.MobileControls;

namespace MSPress.MobWeb.ListItmCmd
{
    public class MyWebForm : System.Web.UI.MobileControls.MobilePage
    {
        protected System.Web.UI.MobileControls.List List1;
        protected System.Web.UI.MobileControls.Label Label4;
        protected System.Web.UI.MobileControls.Form  Form2;

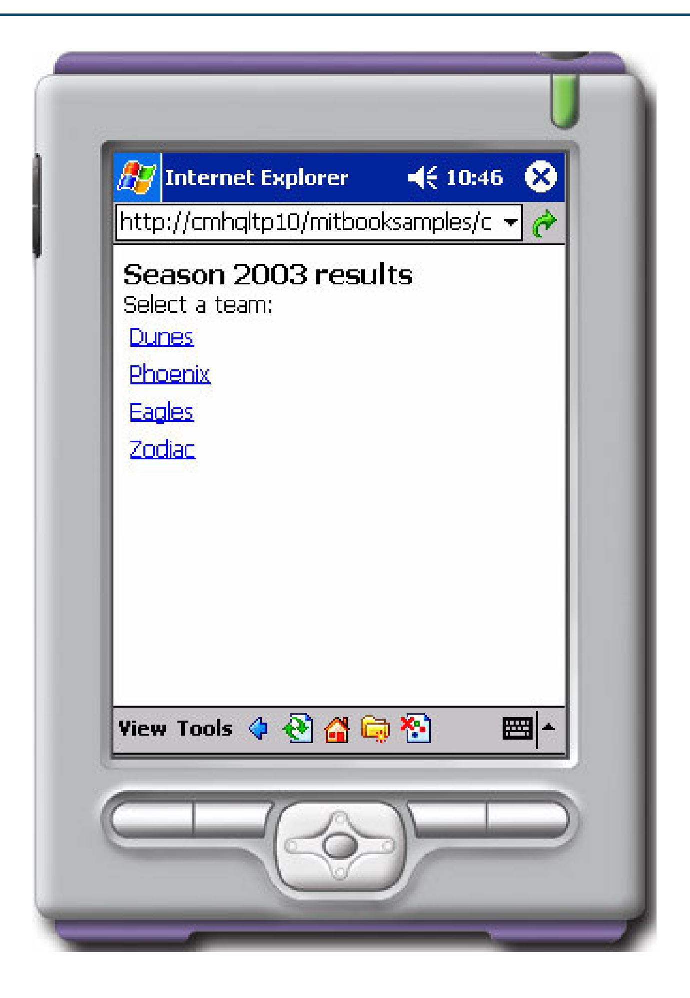
        override protected void OnInit(EventArgs e)
        {
            InitializeComponent();
            base.OnInit(e);
        }

        private void InitializeComponent()
        {
            this.List1.ItemCommand += new
                   ListCommandEventHandler(this.ClickTeamSelection);
        }

        private void ClickTeamSelection(
            Object source,
            ListCommandEventArgs args)
        {
            // Display the Stats page
            this.ActiveForm = Form2;
            String strSelectedTeamStats = args.ListItem.Value;
            Label4.Text = args.ListItem.Text
                           + ": " + strSelectedTeamStats;
        }
    }
}
```

نلاحظ في النص البرمجي أننا قمنا بتحديد العنصر المختار من خلال المعامل args الذي تم تمريره إلى معالج الحدث ClickTeamSelection حيث يمكن الوصول إلى النص والقيمة باستخدام الصيغ args.ListItem.Value و args.ListItem.Text

تكون نتيجة تشغيل هذا المثال على الشكل التالي:



التقسيم إلى صفحات بصورة آلية أومخصصة:

بعكس عنصر التحكم SelectionList يتحدر عنصر التحكم هذا من الصف PagedControl أي أنه يدعم خاصة التقسيم إلى صفحات.

لتفعيل هذه الخاصة يجب إعطاء القيمة True للخاصة Paginate في النموذج الذي يحتوي عنصر التحكم List.

في حال تفعيل التقسيم الآلي إلى صفحات يجري تقسيم خرج القائمة إلى صفحات بحسب قدرة الإظهار الخاصة بالجهاز.

أما في حال استعمال التقسيم المخصص إلى صفحات، فيجب إعطاء قيمة إلى الخاصة ItemCount لإعطاء العدد الكامل من العناصر الذي يمكن إظهاره عبر كل الصفحات.

يقوم عنصر التحكم بالتقسيم إلى صفحات وعند طلب كل صفحة جديدة يجري إطلاق حدث LoadItems.

مثال:

يوضح المثال التالي هذه الفكرة:

```
<%@ Page Inherits="MSPress.MobWeb.CusPag.ExampleWebForm" Language="c#"
    CodeBehind="CustomPaginationExample.aspx.cs" AutoEventWireup=
"False" %>
<%@ Register TagPrefix="mobile"
    Namespace="System.Web.UI.MobileControls"
    Assembly="System.Web.Mobile" %>
```

```
<mobile:Form runat="server" id="Form1" paginate="true">
    <mobile:Label runat="server" StyleReference="title">
        Season 2003 results</mobile:Label>
    <mobile:List id="List1" runat="server"></mobile:List>
</mobile:Form>
```

أما النص البرمجي في الخلفية فهو:

```
using System;
using System.Collections;
using System.Web.UI.MobileControls;

namespace MSPress.MobWeb.CusPag
{
    public class ExampleWebForm :
System.Web.UI.MobileControls.MobilePage
    {
        private TeamStats[] _premierTable;

        protected System.Web.UI.MobileControls.List List1;

        public ExampleWebForm()
        {
            // In the constructor, create the data source we will use.
            _premierTable = new TeamStats[16];
            _premierTable[0] = new TeamStats("Dunes",    "Pts:80");
            _premierTable[1] = new TeamStats("Phoenix",  "Pts:70");
            _premierTable[2] = new TeamStats("Eagles",   "Pts:69");
            _premierTable[3] = new TeamStats("Zodiac",   "Pts:68");
            _premierTable[4] = new TeamStats("Arches",   "Pts:66");
            _premierTable[5] = new TeamStats("Chows",    "Pts:61");
            _premierTable[6] = new TeamStats("Creation", "Pts:57");
            _premierTable[7] = new TeamStats("Illusion", "Pts:54");
            _premierTable[8] = new TeamStats("Torpedo",  "Pts:52");
            _premierTable[9] = new TeamStats("Generals", "Pts:52");
            _premierTable[10] = new TeamStats("Reaction","Pts:51");
            _premierTable[11] = new TeamStats("Peanuts", "Pts:49");
            _premierTable[12] = new TeamStats("Caverns", "Pts:48");
            _premierTable[13] = new TeamStats("Eclipse", "Pts:42");
            _premierTable[14] = new TeamStats("Dragons", "Pts:42");
            _premierTable[15] = new TeamStats("Cosmos",  "Pts:42");
}

        override protected void OnInit(EventArgs e)
        {
            InitializeComponent();
            base.OnInit(e);
        }

        private void InitializeComponent()
        {
```

```
               this.Load += new System.EventHandler(this.Page_Load);
            this.List1.LoadItems +=
                   new LoadItemsEventHandler(this.LoadTeams);
        }
        private void Page_Load(Object sender, EventArgs e)
        {
            // Tell the List how many items it can expect by the time
            // it has asked for them all.
            List1.ItemCount = _premierTable.Length;
        }

        private void LoadTeams(Object source, LoadItemsEventArgs args)
        {
            List1.Items.Clear();
            //  The LoadItemsEventArgs tells us which items and how
many.
            for (int i = 0; i < args.ItemCount; i++)
            {
                // Get the relevant item from the array;
                // Create a MobileListItem.
                int intTablePosn = args.ItemIndex + i;
                MobileListItem lstItem = new MobileListItem(
                    string.Format("{0} {1}",intTablePosn+1,
                        _premierTable[intTablePosn].TeamName),
                   _premierTable[intTablePosn].Stats);

               // Add the item to the Items collection of the List
control.
               List1.Items.Add(lstItem);
            }
        }
    }

    class TeamStats
    {
        private String teamName, stats;

        public TeamStats(String teamName, String stats)
        {
            this.teamName = teamName;
            this.stats = stats;
        }

        public String TeamName
        { get { return this.teamName; } }

        public String Stats
        { get { return this.stats; } }
    }
}
```

تكون نتيجة تنفيذ هذا المثال شبيهة بالشكل التالي:

عناصر تحكم القائمة

عنصر تحكم List

يشبه هذا العنصر عنصر تحكم SelectionList الذي قمنا بمناقشته سابقاً، بفرق أن عنصر التحكم List يدعم عملية تقسيم خرج القائمة إلى عدة صفحات مما يقدم فائدة كبيرة في حالة القوائم الطويلة على الأجهزة المحمولة ذات شاشات الإظهار المحدودة الإمكانيات.

يمكننا استخدام هذا العنصر أيضاً في حالة الرغبة بعرض قوائم تفاعلية وغير تفاعلية. في حالة القوائم التفاعلية لسنا بحاجة إلى زر خاص لعملية إرسال البينات إلى المخدم، إذ يتولى عنصر التحكم هذا عملية إرسال البيانات لدى النقر واختيار أحد العناصر.

يدعم عنصر التحكم List القوالب مما يجعل عنصر التحكم هذا مناسباً في حالة برمجة الاستجابات المتعددة المرتبطة بنوع الجهاز المستخدم.

استخدام عنصر التحكم List:

ذكرنا سابقاً أنه يمكن استخدام هذا العنصر بوضعيتين:

- الوضعية غير التفاعلية: يتم في هذه الوضعية إظهارعناصر القائمة دون القدرة على الاختيار. يمكن هنا تحديد شكل الإظهار باستخدام الخاصة Decoration. ويجب عدم تحديد قيمة لمعالج الحدث ItemCommand في هذه الحالة.
- الوضعية التفاعلية: في حال تم تعيين قيمة لمعالج الحدث ItemCommand يجري تحويل كل عنصر من عناصر القائمة لرابط يقوم باستدعاء معالج الحدث عند النقر عليه.

التقسيم إلى صفحات بصورة آلية أومخصصة:

بعكس عنصر التحكم SelectionList يتحدر عنصر التحكم هذا من الصف PagedControl أي أنه يدعم خاصة التقسيم إلى صفحات.

لتفعيل هذه الخاصة يجب إعطاء القيمة True للخاصة Paginate في النموذج الذي يحتوي عنصر التحكم List.

في حال تفعيل التقسيم الآلي إلى صفحات يجري تقسيم خرج القائمة إلى صفحات بحسب قدرة الإظهار الخاصة بالجهاز.

أما في حال استعمال التقسيم المخصص إلى صفحات، فيجب إعطاء قيمة إلى الخاصة ItemCount لإعطاء العدد الكامل من العناصر الذي يمكن إظهاره عبر كل الصفحات.

يقوم عنصر التحكم بالتقسيم إلى صفحات وعند طلب كل صفحة جديدة يجري إطلاق حدث LoadItems.

عناصر تحكم القائمة

عنصر تحكم ObjectList

يعتبر هذا العنصر، عنصراً شديد المرونة هدفه الأساسي إظهار حقل واحد من مصدر بيانات كقائمة أساسية، وعند اختيار المستخدم أحد عناصر هذه القائمة يقوم بإظهار حقول عديدة أخرى.

بالإضافة إلى ماسبق، يوفر عنصر التحكم ميزة تتعلق بالأوامر التي يمكن ربطها بكل عنصر في القائمة. يمكن أيضاً ربط هذه القائمة بعناصر قائمة أخرى تختلف بحسب العنصر الذي تم النقر عليه. كما هي الحال مع عنصر التحكم List يدعم عنصر التحكم ObjectList القوالب والتقسيم إلى صفحات.

توضح الصيغة التالية استخدام عنصر التحكم هذا:

```
<mobile:ObjectList
   runat="server"
   id="id"
   Alignment="{NotSet|Left|Center|Right}"
   BackColor="backgroundColor"
   Font-Bold="{NotSet|False|True}"
   Font-Italic="{NotSet|False|True}"
   Font-Name="fontName"
   Font-Size="{NotSet|Normal|Small|Large}"
   ForeColor="foregroundColor"
   StyleReference="StyleReference"
   Wrapping="{NotSet|Wrap|NoWrap}"

   AutoGenerateFields="{True|False}"
   CommandStyle="StyleReference"
   DataMember="dataMember"
   DataSource="dataSource"
   DefaultCommand="onDefaultCommandHandler"
   ItemCount="itemCount"
   LabelField="fieldname"
   LabelStyle="StyleReference"
   OnItemDataBind="onItemDataBindHandler"
   OnItemCommand="onItemCommandHandler"
   OnLoadItems="loadItemsHandler">
   OnShowItemCommands="onShowItemCommandsHandler"
   TableFields="tableFields">

   <!-- Optional explicitly declared fields -->
    <Field
        id="id"
        Title="titleText"
        DataField="value"
        FormatString="formatString"
        Visible="{True|False}" />
    </Field>
```

```
    <!-- Optional explicitly declared commands -->
     <Command Name="CommandName" Text="CommandText" />

</mobile:ObjectList>
```

يوضح الجدول التالي أهم الخصائص والأحداث التي يقدمها عنصر التحكم هذا:

الخاصة أو الحدث	النمط	الوصف
AllFields	System.Web.UI. MobileControls. ObjectListFieldCollection	تعيد هذه القيمة غرض ObjectListFieldCollection ObjectListField.لكل حقل يتم إضافته من مصدر البيانات إلى عنصر التحكم ObjectList سواء تم تعريفه باستخدام التأشيرة <Field> أوبواسطة الخاصة Fields ضمن النص البرمجي. يمكنك أن تستخدم خصائص الحقول المحتواة لكن لن تستطيع إضافة أوإزالة حقل من هذه المجموعة
AutoGenerateFields	True|False	تقوم بإعداد عنصر التحكم لإظهار جميع الحقول من غرض DataSet.
BackCommandText	String	تقوم بتعيين أوإعادة النص المستخدم ضمن الرابط الخاص بالعودة من إظهار التفاصيل إلى القائمة الرئيسية.
Commands	System.Web.UI. MobileControls. ObjectList-Command-Collection	تقوم بإعادة غرض ObjectListCommandCollection حيث يوجد غرض ObjectListCommand ضمن هذه المجموعة لكل عنصر يعرف بالتأشيرة <command> يتم إضافته إلى النص البرمجي.
CommandStyle	نمط صالح من الأنماط المستخدمة في StyleSheet	تعيين هذه الخاصة النمط المستخدم لإظهار عناصر الأوامر على الأجهزة. لا يتم تخزين هذه الخاصة تلقائياً بين الطلبات لذلك يجب ضبطها مع كل طلب والطريقة الأفضل لذلك هي تعريفها ضمن صيغة عنصر التحكم من جهة

		المخدم.
DataMember	String	تستخدم فقط عندما يكون عنصر التحكم مرتبطاً بغرض DataSet أوغرض DataTable. تحدد هذه الخاصة اسم الجدول ضمن غرض DataSet الذي سيتم ربط عنصر التحكم به.
DataSource	Object	اسم الغرض DataSet أوغرض Collection الذي يشكل مصدر البيانات
DefaultCommand	String	تظهر القائمة بشكل تلقائي قيمة العنصر المحددة بالخاصة LabelField على شكل ارتباط تشعبي. اختيار هذا الارتباط سيأخذ المستخدم إلى شاشة أخرى تظهر حقول إضافية لهذا العنصر. إذا تم تعيين الخاصة DefaultCommand ضمن النص البرمجي فإن عملية اختيار عنصر من القائمة سيقوم بإطلاق الحدث OnItemCommand مع تمرير قيمة الخاصة هذه إليه. مع العلم أنه سيظل بإمكاننا الوصول إلى تفاصيل العنصر من خلال الوصلة More والتي تظهر بمحاذاة الوصلات إلى أي عناصر أوامر أخرى قمت بتعريفها.
Details	System.Web.UI. MobileControls.Panel	تقوم هذه الخاصة بإعادة عنصر تحكم Panel المستخدم في إظهار تفاصيل العناصر. هذه الخاصة عملية عندما تقوم بتطبيق القوالب وتريد تعيين خصائص عنصر التحكم الذي قمت بوضعه ضمن القالب. ObjectList1.Details .FindControl("ControlID");
DetailsCo mmandText	String	تقوم هذه الخاصة بتعيين أوإعادة النص المستخدم للرابط الذي يقوم بإظهار شاشة التفاصيل. تستخدم هذه الخاصة

مع مستعرضات WML فقط.		
تشبه هذه الخاصة الخاصة AllFields. حيث تقوم بإعادة غرض ObjectListFieldCollection الذي يحتوي غرض ObjectListField لكل حقل في مصدر البيانات تمت إضافته إلى عنصر التحكم ObjectList الذي تم تعريفه باستخدم التأشيرة <Field> أوعن طريق النص البرمجي باستخدام الخاصة Fields. بعكس الخاصة AllFields المعدة للقراءة فقط يمكن بواسطة هذه الخاصة إضافة وإزالة حقول باستخدام طرق الغرض ObjectListFieldCollection	ObjectListFieldCollection	Fields
تستخدم هذه الخاصة في حالة التقسيم المخصص إلى صفحات بحيث تحدد العدد الكامل للعناصر ضمن مصدر البيانات DataSet. لاستخدام هذه الخاصة يجب ألا ننسى ضبط خاصة Form.Paginate إلى القيمة True	Integer	ItemCount
تحدد اسم الحقل في مصدر البيانات الذي سيتم استخدامه كدليل أساسي. حقل الدليل الأساسي يزود القائمة التي ستظهر للمستخدم ويقوم فيها باختيار العنصر لتظهر تفاصيله. تقوم الخاصة LabelFieldIndex بتقديم نفس الفعالية ولكن تحدد الدليل ضمن المجموعة AllFields	String	LabelField
تقوم بإعادة أوتعيين النمط المستخدم لإظهار الترويسة. هذه الخاصة لا تحافظ على قيمتها بين الطلبات المتكررة بذلك فالحل في المحافظة على قيمة ثابتة لها هوتعريفها ضمن صيغة عنصر التحكم	String	LabelStyle

		من جهة المخدم.
MoreText	String	تقوم بتعيين أوإعادة النص المستخدم للرابط More على مستعرضات HTML.
SelectedIndex	Integer	تقوم بإعادة أوتعيين الدليل للعنصر المختار.
Selection	System.Web.UI. MobileControls. ObjectListItem	تقوم بإعادة العنصر المختار أوالقيمة null في حال عدم اختيار أي عنصر.
TableFields	String; قائمة من أسماء الحقول مفصولة بفاصلة منقوطة	إذا لم تقم بتعيين قيمة للخاصة TableFields سيتم إظهار القائمة على أنها مكونة من عمود واحد يتكون فقط من قيم الحقل المحدد بالخاصة LabelField. إذا قمت بتعيين الخاصة DefaultCommand وكان الجهاز يدعم الجداول سيتكون هذا العمود من قيمتين الأولى هي قيم الحقل المحدد بالخاصة LabelField إضافة إلى الرابط More. إذا قمنا بإعطاء قيمة للخاصة TableFields سيتم تمثيل كل عنصر من القائمة باستخدام جدول مع الأعمدة المعرفة بالحقول ضمن هذه الخاصة إضافة إلى حقل تظهر فيه More لتسمح بالوصول إلى رؤية جميع الحقول الخاصة بالعنصر.
ViewMode	System.Web.UI. MobileControls. ObjectListViewMode List|Commands|Details	تسمح هذه الخاصة بتعيين وضع المعاينة للعنصر ObjectList.يمكن أن تأخذ هذه الخاصة القيم: – List التي تقوم بإظهار عناصر القائمة الرئيسية فقط. – Details تقوم بإظهار تفاصيل العنصر المختار. – Commands يتم إظهارها

		فقط على الأجهزة ذات الشاشات الصغيرة كالهواتف المحمولة وهي الشاشة الأولى التي يتم عرضها بعد اختيار المستخدم لعنصر ما حيث تظهر قائمة تتكون من عناصر <Command> إضافة إلى روابط إلى المعاينة التفصيلية. على المستخدم اختيار عنصر من القائمة قبل السماح له بالوصول إلى الخاصة ViewMode
الحدث ItemCommand	اسم طريقة معالج الحدث	يحدد معالج الحدث الذي سيتم استدعاؤه عندما يقوم المستخدم باختيار عنصر من الأوامر المرتبطة بإظهار تفاصيل أحد العناصر. يتم تعريف الأوامر باستخدام التأشيرة <Command> أومن خلال التعامل مع المجموعة التي تحددها الخاصة Commands
الحدث LoadItems	اسم طريقة معالج الحدث	هذا الحدث مطلوب حين يتم تعيين الخاصة ItemCount لتفعيل التقسيم المخصص إلى صفحات. يقوم التطبيق باستدعاء معالج الحدث هذا في كل مرة يتم طلب بيانات جديدة. يسمح هذا الحدث بتمرير البيانات إلى عنصر التحكم حسب الطلب. عوضاً عن تمرير كل البيانات دفعة واحدة.
الحدث ShowItemCommands	اسم طريقة معالج الحدث	يحدد معالج الحدث الذي يتم استدعاؤه عندما يتم إظهار التفاصيل عنصر وتشكيل الوصلات الخاصة بالأوامر. في معالج الحدث هذا يمكن إضافة أوحذف أوامر أوبناء قائمة من الأوامر للعنصر الذي يتم إظهاره.

استخدام عنصر التحكم ObjectList:

يمكن عنصر التحكم هذا من أداء مجموعة من الأعمال تفوق ما يمكن أن يؤديه عنصر التحكم List.

أهم الوظائف التي يمكن لعنصر التحكم هذا القيام بها ولا يمكن تأديتها من خلال عنصر تحكم List هي:

1- إظهار عدة حقول من مصدر بيانات.
2- إظهار العناصر ضمن جدول عوضاً عن إظهارها كقائمة من عمود واحد.
3- تزويد أكثر من أمر مرتبط بنفس العنصر من القائمة.
4- تزويد عدة أوامر للخيارات المتعددة من القائمة.
5- إظهار حقل وحيد مع إمكانية إظهار عدة حقول كوظيفة ثانوية.

لتوضيح كل من هذه الميزات سنقوم باستخدامها ضمن مثال خاص:

1- إظهار عدة حقول من مصدر بيانات:

```
<%@ Page Inherits="MSPress.MobWeb.ObjListEx.MyWebForm" Language="c#"
    CodeBehind="ObjectListExample.aspx.cs" AutoEventWireup="False"  %>
<%@ Register TagPrefix="mobile"
    Namespace="System.Web.UI.MobileControls"
    Assembly="System.Web.Mobile" %>

<mobile:Form runat="server" >
    <mobile:Label runat="server" StyleReference="title">
        Season 2003 results</mobile:Label>
    <mobile:ObjectList id="ObjectList1" runat="server">
        AutoGenerateFields="false">
        <Field Title="Team" DataField="TeamName"></Field>
        <Field Title="Won" DataField="Won"></Field>
        <Field Title="Drawn" DataField="Drawn"></Field>
        <Field Title="Lost" DataField="Lost"></Field>
        <Field Title="Pts" DataField="Points" Visible="false"></Field>
    </mobile:ObjectList>
</mobile:Form>
```

أما النص البرمجي في الخلفية فهو:

```
using System;
using System.Collections;
using System.Web.UI.MobileControls;

namespace MSPress.MobWeb.ObjListEx
{
    public class MyWebForm : System.Web.UI.MobileControls.MobilePage
    {
        protected ObjectList ObjectList1;

        override protected void OnInit(EventArgs e)
```

```
        {
            InitializeComponent();
            base.OnInit(e);
        }

        private void InitializeComponent()
        {
                this.Load += new System.EventHandler(this.Page_Load);
        }

        private void Page_Load(Object sender, EventArgs e)
        {
            if (!IsPostBack)
            {
                ArrayList array = new ArrayList();
                array.Add(new TeamStats("Dunes",1,38,24,8,6,80));
                array.Add(new TeamStats("Phoenix",2,38,20,10,8,70));
                array.Add(new TeamStats("Eagles",3,38,20,9,9,69));
                array.Add(new TeamStats("Zodiac",4,38,20,8,10,68));

                ObjectList1.DataSource = array;
                ObjectList1.LabelField = "TeamName";
                ObjectList1.DataBind();
            }
        }
    }

    class TeamStats
    {
        private String  _teamName;
        private int _position, _played, _won, _drawn, _lost, _points;

        public TeamStats(String teamName,
            int position,
            int played,
            int won,
            int drawn,
            int lost,
            int points)
        {
            this._teamName = teamName;
            this._position = position;
            this._played = played;
            this._won = won;
            this._drawn = drawn;
            this._lost = lost;
            this._points = points;
        }

        public String TeamName { get { return this._teamName; } }
        public int    Position { get { return this._position; } }
        public int    Played   { get { return this._played; } }
```

```
        public int    Won      { get { return this._won; } }
        public int    Drawn    { get { return this._drawn; } }
        public int    Lost     { get { return this._lost; } }
        public int    Points   { get { return this._points; } }
    }
}
```

2- إظهار العناصر ضمن جدول عوضاً عن إظهارها كقائمة من عمود واحد.

```
<%@ Page Inherits="MSPress.MobWeb.ObjListTblEx.MyWebForm" Language="c#"
    CodeBehind="ObjectListTableExample.aspx.cs" AutoEventWireup="False"
%>
<%@ Register TagPrefix="mobile"
Namespace="System.Web.UI.MobileControls"
    Assembly="System.Web.Mobile" %>

<mobile:Form runat="server"
    <mobile:Label runat="server" StyleReference="title">
        Season 2003 results</mobile:Label>
    <mobile:ObjectList id="ObjectList1"
                       runat="server"
                       AutoGenerateFields="true"
                       TableFields="TeamName;Position;Points">
    </mobile:ObjectList>
</mobile:Form>
```

3- تزويد أكثر من أمر مرتبط بنفس العنصر من القائمة.

```
<%@ Page Inherits="MSPress.MobWeb.ObjListCmdsEx.MyWebForm"
Language="c#"
    CodeBehind="ObjectListItemCommandsExample.aspx.cs"
    AutoEventWireup="False" %>
<%@ Register TagPrefix="mobile"
    Namespace="System.Web.UI.MobileControls"
    Assembly="System.Web.Mobile" %>

<mobile:Form runat="server" id="Form1">
    <mobile:Label runat="server" StyleReference="title">
        Season 2003 results</mobile:Label>
    <mobile:ObjectList id="ObjectList1" runat="server"
                       AutoGenerateFields="false"
                       LabelField="TeamName">
        <Field Title="Team" DataField="TeamName"></Field>
        <Field Title="Won" DataField="Won"></Field>
        <Field Title="Drawn" DataField="Drawn"></Field>
        <Field Title="Lost" DataField="Lost"></Field>
        <Field Title="Points" DataField="Points"></Field>
```

```
        <Field Title="Champs. Cup" DataField="ChampionsCup"
               Visible="false">
        </Field>
        <Field Title="Inter-City Cup" DataField="InterCup"
Visible="false">
        </Field>
        <Command Name="ChampsCup" Text="Champions Cup"/>
        <Command Name="InterCityCup" Text="Inter-City Cup"/>
    </mobile:ObjectList>
</mobile:Form>

<mobile:Form runat="server" id="Form2">
    <mobile:Label runat="server" StyleReference="title">
        Season 2003 European Results</mobile:Label>
    <mobile:Label runat="server" id="Label1"/>
    <mobile:Link runat="server" NavigateUrl="#Form1">
        Back
    </mobile:Link>
</mobile:Form>
```

أما النص البرمجي في الخلفية فهو:

```
using System;
using System.Collections;
using System.Web.UI.MobileControls;

namespace MSPress.MobWeb.ObjListCmdsEx
{
    public class MyWebForm : System.Web.UI.MobileControls.MobilePage
    {
        protected ObjectList ObjectList1;
        protected Form       Form2;
        protected Label      Label1;

        override protected void OnInit(EventArgs e)
        {
            InitializeComponent();
            base.OnInit(e);
        }

        private void InitializeComponent()
        {
            this.Load += new System.EventHandler(this.Page_Load);
            this.ObjectList1.ItemCommand +=
                new
ObjectListCommandEventHandler(this.Team_OnItemCommand);
        }

        private void Page_Load(Object sender, EventArgs e)
        {
```

```
        if (!IsPostBack)
        {
            ArrayList array = new ArrayList();
            array.Add(new TeamStats("Dunes",1,38,24,8,6,80,
                                        "Quarter Finals",""));
            array.Add(new TeamStats("Phoenix",2,38,20,10,8,70,
                                        "Quarter Finals",""));
            array.Add(new TeamStats("Eagles",3,38,20,9,9,69,
                                        "","Winners"));
            array.Add(new TeamStats("Zodiac",4,38,20,8,10,68,
                                        "Semi Finals",""));

            ObjectList1.DataSource = array;
            ObjectList1.LabelField = "TeamName";
            ObjectList1.DataBind();
        }
    }

    private void Team_OnItemCommand(
        Object sender,
        ObjectListCommandEventArgs e)
    {
        Label1.Text = "Did Not Compete"; //Default
        this.ActiveForm = Form2;

        if (e.CommandName == "ChampsCup")
        {
            // Set the label to the Champions Cup result.
            if (e.ListItem["ChampionsCup"] != "")
                Label1.Text = "Champions Cup: " +
                    e.ListItem["ChampionsCup"];
        }
        else if (e.CommandName == "InterCityCup")
        {
            // Set the label to the Inter-City Cup result.
            if (e.ListItem["InterCup"] != "")
                Label1.Text = " Inter-City Cup: " +
                    e.ListItem["InterCup"];
        }
    }
}

class TeamStats
{
    private String  _teamName;
    private int _position, _played, _won, _drawn, _lost, _points;
    private String  _champsCup, _interCup;

    public TeamStats(String teamName,
        int position,
        int played,
        int won,
```

```
            int drawn,
            int lost,
            int points,
            String championsCup,
            String interCup)
        {
            this._teamName = teamName;
            this._position = position;
            this._played = played;
            this._won = won;
            this._drawn = drawn;
            this._lost = lost;
            this._points = points;
            this._champsCup = championsCup;
            this._interCup= interCup;
        }

        public String TeamName { get { return this._teamName; }}
        public int    Position { get { return this._position; }}
        public int    Played   { get { return this._played; }}
        public int    Won      { get { return this._won; }}
        public int    Drawn    { get { return this._drawn; }}
        public int    Lost     { get { return this._lost; }}
        public int    Points   { get { return this._points; }}
        public String ChampionsCup { get { return this._champsCup; }}
        public String InterCup{ get { return this._interCup; }}
    }
}
```

4- تزويد عدة أوامر للخيارات المتعددة من القائمة.

```
<%@ Page Inherits="MSPress.MobWeb.ObjListShowItems.MyWebForm"
Language="c#"
    CodeBehind="ObjectListOnShowItemsExample.aspx.cs"
    AutoEventWireup="False" %>
<%@ Register TagPrefix="mobile"
    Namespace="System.Web.UI.MobileControls"
    Assembly="System.Web.Mobile" %>

<mobile:Form runat="server" id="Form1">
    <mobile:Label runat="server" StyleReference="title">
        Season 2003 results</mobile:Label>
    <mobile:ObjectList id="ObjectList1" runat="server"
                       AutoGenerateFields="true"
                       LabelField="TeamName">
        <Command Name="ChampsCup" Text="Champions Cup"/>
        <Command Name="InterCityCup" Text="Inter-City Cup"/>
    </mobile:ObjectList>
</mobile:Form>
```

```
<mobile:Form runat="server" id="Form2">
    <mobile:Label runat="server" StyleReference="title" id="Label1"/>
    <mobile:Label runat="server" id="Label2"/>
    <mobile:Link runat="server" NavigateUrl="#Form1">
        Back
    </mobile:Link>
</mobile:Form>
```

والنص البرمجي في الخلفية يأخذ الشكل:

```
using System;
using System.Collections;
using System.Web.UI.MobileControls;

namespace MSPress.MobWeb.ObjListShowItems
{
    public class MyWebForm : System.Web.UI.MobileControls.MobilePage
    {
        protected ObjectList ObjectList1;
        protected Form       Form2;
        protected Label      Label1;
        protected Label      Label2;

        override protected void OnInit(EventArgs e)
        {
            InitializeComponent();
            base.OnInit(e);
        }

        private void InitializeComponent()
        {
            this.Load += new System.EventHandler(this.Page_Load);
            this.ObjectList1.ItemCommand += new

ObjectListCommandEventHandler(this.Team_OnItemCommand);
            this.ObjectList1.ShowItemCommands += new

ObjectListShowCommandsEventHandler(this.SetItemCommands);
        }

        private void Page_Load(Object sender, EventArgs e)
        {
            // Not shown
            // As in previous example
        }

        private void Team_OnItemCommand(
            Object sender,
```

```
            ObjectListCommandEventArgs e)
        {
            this.ActiveForm = Form2;

            if (e.CommandName == "ChampsCup")
            {
                // Set the label to the Champions Cup result.
                Label1.Text = "Champions Cup 2003";
                Label2.Text = e.ListItem["ChampionsCup"];
            }
            else if (e.CommandName == "InterCityCup")
            {
                // Set the label to the Inter-City Cup result.
                Label1.Text = "Inter-City Cup 2003";
                Label2.Text = e.ListItem["InterCup"];
            }
        }

        private void SetItemCommands(
            Object sender,
            ObjectListShowCommandsEventArgs  e)
        {
            // Remove either the Champions Cup or Inter-City Cup
            // command if the team didn't compete (if field is blank).
            if (e.ListItem["ChampionsCup"] == "")
                e.Commands.Remove("ChampsCup");

            if (e.ListItem["InterCup"] == "")
                e.Commands.Remove("InterCityCup");
        }
    }

    class TeamStats
    {
        private String  _teamName;
        private int _position, _played, _won, _drawn, _lost, _points;
        private String  _champsCup, _interCup;

        public TeamStats(String teamName,
            int position,
            int played,
            int won,
            int drawn,
            int lost,
            int points,
            String championsCup,
            String interCup)
        {
            this._teamName = teamName;
            this._position = position;
            this._played = played;
            this._won = won;
```

```
            this._drawn = drawn;
            this._lost = lost;
            this._points = points;
            this._champsCup = championsCup;
            this._interCup= interCup;
        }

        public String TeamName { get { return this._teamName; }}
        public int    Position { get { return this._position; }}
        public int    Played   { get { return this._played; }}
        public int    Won      { get { return this._won; }}
        public int    Drawn    { get { return this._drawn; }}
        public int    Lost     { get { return this._lost; }}
        public int    Points   { get { return this._points; }}
        public String ChampionsCup { get { return this._champsCup; }}
        public String InterCup{ get { return this._interCup; }}
    }
}
```

عنصر تحكم ObjectList:

عنصر التحكم هذا هوعنصر شديد المرونة هدفه الأساسي إظهار حقل واحد من مصدر بيانات كقائمة أساسية وعند اختيار المستخدم أحد عناصر هذه القائمة سيقوم بإظهار حقول عديدة أخرى.

بالإضافة إلى هذا يوفر عنصر التحكم ميزة تتعلق بالأوامر التي يمكن ربطها بكل عنصر في القائمة. يمكن أيضاً ربط هذه القائمة بعناصر قائمة أخرى تختلف بحسب العنصر الذي تم النقر عليه.

كما هي الحال مع عنصر التحكم List يدعم عنصر التحكم ObjectList القوالب والتقسيم إلى صفحات.

القسم التاسع والعاشر:

الموضوع الثالث: عناصر التحكم المُخصَصَة

الكلمات المفتاحية:

عنصر تحكم، محمول، لغة تأشير، تأشيرة، واصفة، خاصة، حدث.

ملخص:

سنتعرف في هذه الجلسة على كيفية التعامل مع بعض عناصر التحكم المخصصة وعلى كيفية إعدادها وأهم الميزات التي تقدمها.
ستغطي هذه الجلسة عناصر تحكم MobileCheckBox، MobileMultiLineInput، MobileDynamicImageControl

أهداف تعليمية:

يتعرف الطالب في هذا الفصل على:

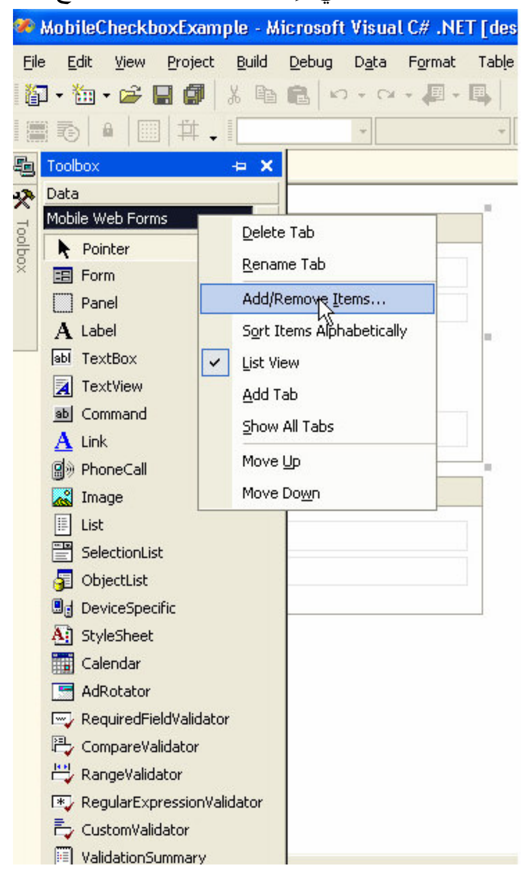
- عنصر التحكم MobileCheckBox
- عنصر التحكم MobileMultiLineInput
- عنصر التحكم MobileDynamicImageControl.

استخدام عناصر التحكم المخصصة

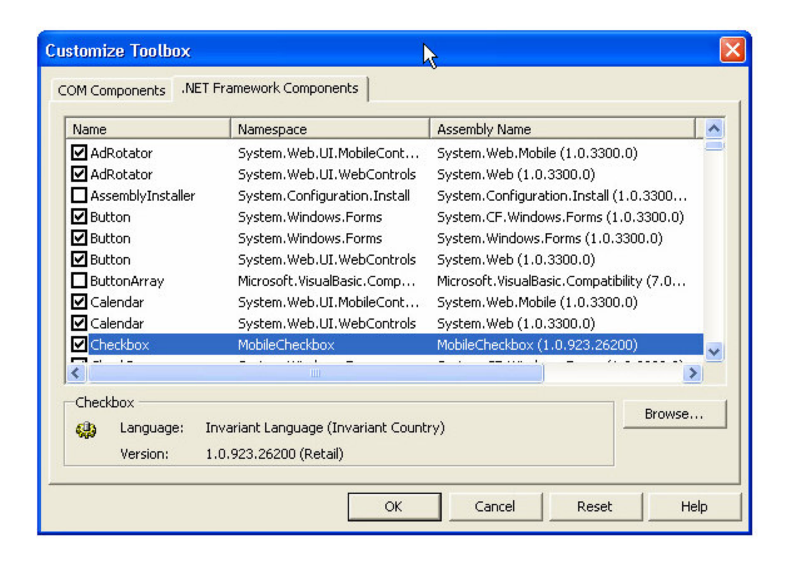
تتضمن الكثير من المواقع ومنها http: //www.asp.net العديد من عناصر التحكم التي تم بناؤها من قبل مطورين، والتي لا تنتمي إلى مجموعة عناصر التحكم التي تقدمها ASP.NET Mobile فكيف سنقوم باستخدام هذه العناصر ضمن تطبيقاتنا.

في حال استعمال Visual Studio:

في حال استعمال بيئة التطوير الخاصة Visual Studio يكفي لإضافة هذه العناصر اتباع المراحل التالية:



ثم قم باختيار زر Browse وانتقاء الملف الذي يعبر عن النص المفسر لعنصر التحكم



أما في حالة عدم استخدام بيئة التطوير Visual Studio :

يجب في هذه الحالة نسخ الملفات التجميعية ضمن المجلد bin/ الخاص بالتطبيق ومن ثم إضافة المرجع إلى بداية ملف aspx.

فمثلاً في حالة عنصر التحكم MobileCheckBox يكون التعبير المضاف من الشكل:

```
<%@ Register TagPrefix="mobCB" Namespace="MobileCheckbox"
    Assembly="MobileCheckbox" %>
```

وبعدها يمكننا استخدام عنصر التحكم هذا بالصيغة

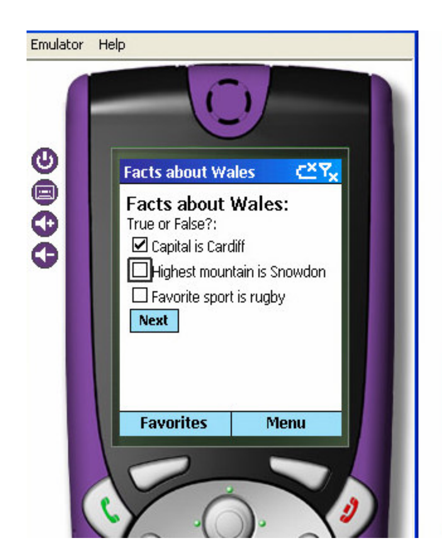
```
<mobCB:Checkbox id="Checkbox1" runat="server" …></mobCB:Checkbox>
```

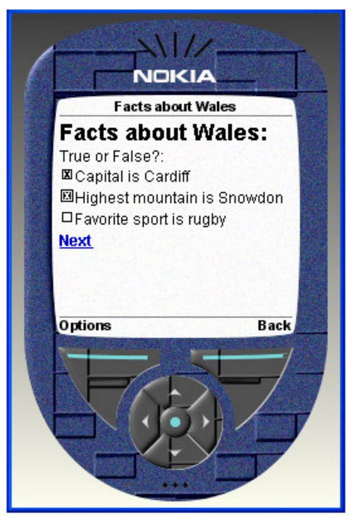
–

عنصر التحكم MobileCheckBox

يقدم عنصر التحكم هذا آلية عنصر التحكم CheckBox العادية مع إمكانية خاصة للعمل على مستعرضات HTML، CHTML وWML.

يبين الشكل التالي كيف يظهر عنصر التحكم هذا على المستعرضات المختلفة.





للاستفادة من عنصر التحكم هذا لا بد من إعداد تطبيقك من خلال إضافة <device> ضمن تأشيرة <mobileControls> في ملف الإعداد web.config:

```
<configuration>
  <system.web>
    <mobileControls>

      <device name="MobileCheckboxHtmlDeviceAdapter"
          inheritsFrom="HtmlDeviceAdapters">
        <control name="MobileCheckbox.Checkbox,MobileCheckbox"
          adapter="MobileCheckbox.HtmlCheckboxAdapter,MobileCheckbox"/>
      </device>
      <device name="MobileCheckboxWmlDeviceAdapter"
          inheritsFrom="WmlDeviceAdapters">
        <control name="MobileCheckbox.Checkbox,MobileCheckbox"
          adapter="MobileCheckbox.WmlCheckboxAdapter,MobileCheckbox"/>
      </device>
      <device name="MobileCheckboxChtmlDeviceAdapter"
          inheritsFrom="ChtmlDeviceAdapters">
        <control name="MobileCheckbox.Checkbox,MobileCheckbox"

adapter="MobileCheckbox.ChtmlCheckboxAdapter,MobileCheckbox"/>
      </device>

    </mobileControls>
  </system.web>
</configuration>
```

يستخدم عنصر التحكم هذا الصيغة التالية:

```
<%@ Register TagPrefix="mobCB" Namespace="MobileCheckbox"
    Assembly="MobileCheckbox" %>

<mobCB:Checkbox
   runat="server"
```

```
    id="id"
    Alignment="{NotSet|Left|Center|Right}"
    BackColor="backgroundColor"
    BreakAfter="{True|False}"
    Font-Bold="{NotSet|False|True}"
    Font-Italic="{NotSet|False|True}"
    Font-Name="fontName"
    Font-Size="{NotSet|Normal|Small|Large}"
    ForeColor="foregroundColor"
    StyleReference="StyleReference"
    Wrapping="{NotSet|Wrap|NoWrap}"

    AutoPostBack="{True|False}"
    Checked="{True|False}"
    OnCheckedChanged="EventHandlerMethodName"
    TextAlign="{Left|Right}"
    Text="LabelText"
</mobCB:Checkbox>
```

يبين الجدول التالي يبين الخصائص والأحداث التي يدعمها عنصر التحكم هذا:

الخاصة أو الحدث	النمط	الوصف
AutoPostBack	True|False	تقوم بتعيين أوإعادة القيمة التي تحدد فيما إذا كان سيجري إرسال حالة عنصر التحكم إلى المخدم عند النقر على هذاالعنصر.تكون القيمة التلقائية لهذه الخاصة هي False ويمكن تطبيقها على مستعرضات Html فقط. في حال كانت لهذه الخاصة القيمة False نحتاج إلى وجود عنصر آخر ليقوم بعملية إرسال النموذج إلى المخدم.
Checked	True | False	تقوم بتعيين أوإعادة القيمة التي تحدد فيما إذا كان عنصر التحكم مختاراً في الوضع البدائي.
TextAlign	System.Web.UI.WebControls.TextAlign Left|Right	يتم استخدام هذه الخاصة لتحديد محاذاة النص المرتبط بعنصر التحكم.
Text	String	تستخدم هذه الخاصة لتعيين أوإعادة قيمة النص المرتبط مع عنصر

		التحكم هذا.
الحدث CheckedChanged	طريقة معالج الحدث	تعيين اسم طريقة معالج الحدث حيث يتم إطلاق هذا الحدث عند تغيير قيمة الخاصة Checked.

المثال التالي يوضح استخدام عنصر التحكم هذا:

```
<%@ Register TagPrefix="cc1" Namespace="MobileCheckbox"
    Assembly="MobileCheckbox" %>
<%@ Page language="c#" Codebehind="default.aspx.cs"
    Inherits="MSPress.MobWeb.CheckboxEx._default"
    AutoEventWireup="false" %>
<%@ Register TagPrefix="mobile"
Namespace="System.Web.UI.MobileControls"
    Assembly="System.Web.Mobile" %>

<mobile:Form id="Form1" runat="server" title="Facts about Wales">
    <mobile:Label id="Label3" runat="server" StyleReference="title">
        Facts about Wales:</mobile:Label>
    <mobile:Label id="Label2" runat="server">
        True or False?:</mobile:Label>
    <cc1:Checkbox id="Checkbox1" runat="server"
        Text="Capital is Cardiff"></cc1:Checkbox>
    <cc1:Checkbox id="Checkbox2" runat="server"
        Text="Highest mountain is Snowdon"></cc1:Checkbox>
    <cc1:Checkbox id="Checkbox3" runat="server"
        Text="Favorite sport is rugby"></cc1:Checkbox>
    <mobile:Command id="Command1" runat="server">Next</mobile:Command>
</mobile:Form>

<mobile:Form id="Form2" runat="server" title="Result">
    <mobile:Label id="Label1" runat="server"
        StyleReference="title">Result</mobile:Label>
    <mobile:Label id="result" runat="server">Label</mobile:Label>
</mobile:Form>
```

أما النص البرمجي في الخلفية فهو:

```
using System;
using System.Web.UI.MobileControls;

namespace MSPress.MobWeb.CheckboxEx
{
    public class _default : System.Web.UI.MobileControls.MobilePage
    {
        protected MobileCheckbox.Checkbox Checkbox1;
        protected System.Web.UI.MobileControls.Command Command1;
        protected System.Web.UI.MobileControls.Form Form2;
        protected MobileCheckbox.Checkbox Checkbox2;
        protected MobileCheckbox.Checkbox Checkbox3;
```

```
        protected System.Web.UI.MobileControls.Label result;
        protected System.Web.UI.MobileControls.Form Form1;

        override protected void OnInit(EventArgs e)
        {
            InitializeComponent();
            base.OnInit(e);
        }

        private void InitializeComponent()
        {
            this.Command1.Click +=
                new System.EventHandler(this.Command1_Click);
        }

        private void Command1_Click(object sender, System.EventArgs e)
        {
            int correct = 0;
            if (Checkbox1.Checked) correct++;
            if (Checkbox2.Checked) correct++;
            if (Checkbox3.Checked) correct++;

            result.Text = "You got " + correct.ToString() + " correct";

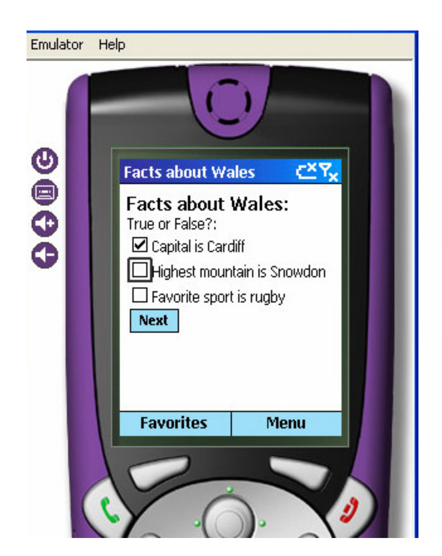
            ActiveForm = Form2;
        }
    }
}
```

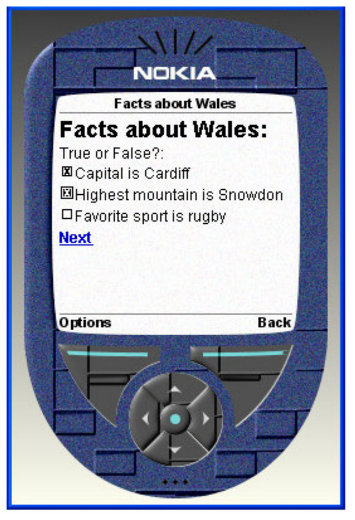
يقدم عنصر التحكم هذا آلية عنصر التحكم CheckBox العادية مع إمكانية خاصة للعمل على مستعرضات HTML، CHTML وWML.

عنصر التحكم MobileCheckBox

يقدم عنصر التحكم هذا آلية عنصر التحكم CheckBox العادية مع إمكانية خاصة للعمل على مستعرضات HTML، CHTML وWML.

يبين الشكل التالي كيف يظهر عنصر التحكم هذا على المستعرضات المختلفة.





للاستفادة من عنصر التحكم هذا لا بد من إعداد تطبيقك من خلال إضافة <device> ضمن تأشيرة <mobileControls> في ملف الإعداد web.config:

```
<configuration>
  <system.web>
    <mobileControls>

      <device name="MobileCheckboxHtmlDeviceAdapter"
          inheritsFrom="HtmlDeviceAdapters">
        <control name="MobileCheckbox.Checkbox,MobileCheckbox"
          adapter="MobileCheckbox.HtmlCheckboxAdapter,MobileCheckbox"/>
      </device>
      <device name="MobileCheckboxWmlDeviceAdapter"
          inheritsFrom="WmlDeviceAdapters">
        <control name="MobileCheckbox.Checkbox,MobileCheckbox"
          adapter="MobileCheckbox.WmlCheckboxAdapter,MobileCheckbox"/>
      </device>
      <device name="MobileCheckboxChtmlDeviceAdapter"
          inheritsFrom="ChtmlDeviceAdapters">
        <control name="MobileCheckbox.Checkbox,MobileCheckbox"

adapter="MobileCheckbox.ChtmlCheckboxAdapter,MobileCheckbox"/>
      </device>

    </mobileControls>
  </system.web>
</configuration>
```

يستخدم عنصر التحكم هذا الصيغة التالية:

```
<%@ Register TagPrefix="mobCB" Namespace="MobileCheckbox"
    Assembly="MobileCheckbox" %>

<mobCB:Checkbox
   runat="server"
   id="id"
   Alignment="{NotSet|Left|Center|Right}"
   BackColor="backgroundColor"
   BreakAfter="{True|False}"
   Font-Bold="{NotSet|False|True}"
   Font-Italic="{NotSet|False|True}"
   Font-Name="fontName"
   Font-Size="{NotSet|Normal|Small|Large}"
   ForeColor="foregroundColor"
   StyleReference="StyleReference"
   Wrapping="{NotSet|Wrap|NoWrap}"

   AutoPostBack="{True|False}"
   Checked="{True|False}"
   OnCheckedChanged="EventHandlerMethodName"
   TextAlign="{Left|Right}"
   Text="LabelText"
</mobCB:Checkbox>
```

يبين الجدول التالي يبين الخصائص والأحداث التي يدعمها عنصر التحكم هذا:

الخاصة أو الحدث	النمط	الوصف
AutoPostBack	True|False	تقوم بتعيين أوإعادة القيمة التي تحدد فيما إذا كان سيجري إرسال حالة عنصر التحكم إلى المخدم عند النقر على هذاالعنصر.تكون القيمة التلقائية لهذه الخاصة هي False ويمكن تطبيقها على مستعرضات Html فقط. في حال كانت لهذه الخاصة القيمة False نحتاج إلى وجود عنصر آخر ليقوم بعملية إرسال النموذج إلى المخدم.
Checked	True | False	تقوم بتعيين أوإعادة القيمة التي تحدد فيما إذا كان عنصر التحكم مختاراً في الوضع البدائي.
TextAlign	System.Web.UI.WebControls.TextAlign Left|Right	يتم استخدام هذه الخاصة لتحديد محاذاة النص المرتبط بعنصر التحكم.
Text	String	تستخدم هذه الخاصة لتعيين أوإعادة قيمة النص المرتبط مع عنصر التحكم هذا.
الحدث CheckedChanged	طريقة معالج الحدث	تعيين اسم طريقة معالج الحدث. يتم إطلاق هذا الحدث عند تغيير قيمة الخاصة Checked.

يوضح المثال التالي استخدام عنصر التحكم هذا:

```
<%@ Register TagPrefix="cc1" Namespace="MobileCheckbox"
    Assembly="MobileCheckbox" %>
<%@ Page language="c#" Codebehind="default.aspx.cs"
    Inherits="MSPress.MobWeb.CheckboxEx._default"
    AutoEventWireup="false" %>
<%@ Register TagPrefix="mobile"
Namespace="System.Web.UI.MobileControls"
    Assembly="System.Web.Mobile" %>

<mobile:Form id="Form1" runat="server" title="Facts about Wales">
    <mobile:Label id="Label3" runat="server" StyleReference="title">
```

```
        Facts about Wales:</mobile:Label>
    <mobile:Label id="Label2" runat="server">
        True or False?:</mobile:Label>
    <cc1:Checkbox id="Checkbox1" runat="server"
        Text="Capital is Cardiff"></cc1:Checkbox>
    <cc1:Checkbox id="Checkbox2" runat="server"
        Text="Highest mountain is Snowdon"></cc1:Checkbox>
    <cc1:Checkbox id="Checkbox3" runat="server"
        Text="Favorite sport is rugby"></cc1:Checkbox>
    <mobile:Command id="Command1" runat="server">Next</mobile:Command>
</mobile:Form>

<mobile:Form id="Form2" runat="server" title="Result">
    <mobile:Label id="Label1" runat="server"
        StyleReference="title">Result</mobile:Label>
    <mobile:Label id="result" runat="server">Label</mobile:Label>
</mobile:Form>
```

أما النص البرمجي في الخلفية فهو:

```
using System;
using System.Web.UI.MobileControls;

namespace MSPress.MobWeb.CheckboxEx
{
    public class _default : System.Web.UI.MobileControls.MobilePage
    {
        protected MobileCheckbox.Checkbox Checkbox1;
        protected System.Web.UI.MobileControls.Command Command1;
        protected System.Web.UI.MobileControls.Form Form2;
        protected MobileCheckbox.Checkbox Checkbox2;
        protected MobileCheckbox.Checkbox Checkbox3;
        protected System.Web.UI.MobileControls.Label result;
        protected System.Web.UI.MobileControls.Form Form1;

        override protected void OnInit(EventArgs e)
        {
            InitializeComponent();
            base.OnInit(e);
        }

        private void InitializeComponent()
        {
            this.Command1.Click +=
                new System.EventHandler(this.Command1_Click);
        }

        private void Command1_Click(object sender, System.EventArgs e)
        {
            int correct = 0;
            if (Checkbox1.Checked) correct++;
            if (Checkbox2.Checked) correct++;
```

```
            if (Checkbox3.Checked) correct++;

            result.Text = "You got " + correct.ToString() + " correct";

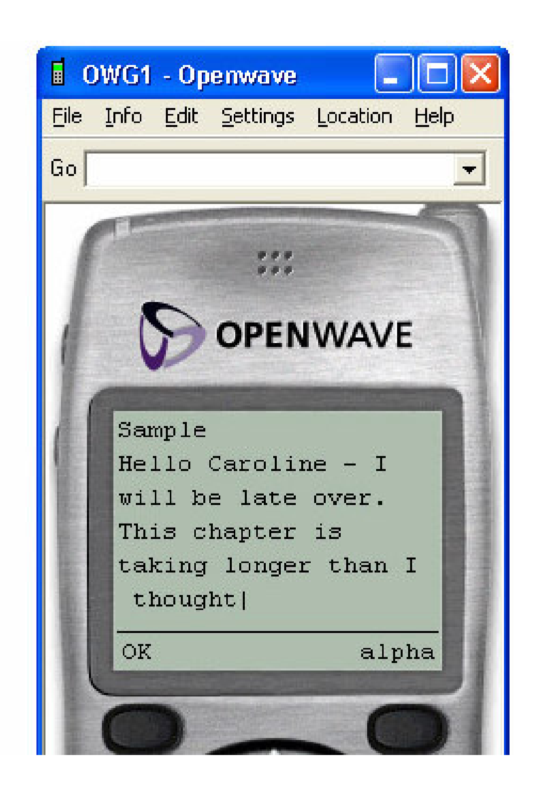
            ActiveForm = Form2;
        }
    }
}
```

يقدم عنصر التحكم هذا آلية عنصر التحكم CheckBox العادية مع إمكانية خاصة للعمل على مستعرضات HTML، CHTML و WML.

عنصر التحكم MobileMultiLineInput

يقدم هذا العنصر وظيفة إدخال نص متعدد الأسطر كأحد عناصر التحكم المحمولة. تستطيع التطبيقات التي تتطلب كمية كبيرة من الدخل النصي كتطبيقات الرسائل والحوار الاستفادة من عنصر التحكم هذا.





إعداد التطبيق:

كما ذكرنا أنه لا بد لنا من إعداد التطبيق من خلال ضبط الملف Web.config

```
<configuration>
    <system.web>
        <compilation debug="true">
            <assemblies>
                <add assembly="MLIC" />
            </assemblies>
        </compilation>
        <mobileControls>
            <device name="MMITTextInputHtmlDeviceAdapter"
                inheritsFrom="HtmlDeviceAdapters">
                <control name="MMIT_Sample.MultiLineInput,MLIC"

adapter="MMIT_Sample.HtmlMultiLineInputAdapter,MLIC"/>
            </device>
        </mobileControls>
    </system.web>
</configuration>
```

أما الصيغة التي يستخدمها عنصر التحكم هذا فهي التالية:

```
<%@ Register TagPrefix="mobMLI" Namespace="MMIT_Sample"
    Assembly="MLIC" %>

<mobMLI:MultiLineInput
    runat="server"
    id="id"
    Alignment="{NotSet|Left|Center|Right}"
    BackColor="backgroundColor"
    Font-Bold="{NotSet|False|True}"
    Font-Italic="{NotSet|False|True}"
    Font-Name="fontName"
    Font-Size="{NotSet|Normal|Small|Large}"
    ForeColor="foregroundColor"
    StyleReference="StyleReference"
    Wrapping="{NotSet|Wrap|NoWrap}"

    MaxLength="maxlength"
    Numeric="{True|False}"
    Password="{True|False}"
    OnTextChanged="textChangedEventHandler"
    Size="textBoxLength"
    Text="Text"
    Title="Text"

    Rows="{number of rows}"
    Cols="{number of columns}" >

</mobMLI:MultiLineInput>
```

كما نلاحظ يوفر عنصر التحكم هذا خاصتين أساسيتين:

الخاصة	النمط	الوصف
Rows	Integer	تحدد هذه الخاصة عدد الصفوف التي سيتم إظهارها لهذا العنصر. يتم تطبيق هذه الخاصة على مستعرضات HTML فقط
Cols	Integer	تحدد هذه الخاصة عدد الأعمدة التي سيتم إظهارها لهذا العنصر.

المثال التالي يوضح استخدام عنصر التحكم هذا:

```
<%@ Register TagPrefix="cc1" Namespace="MMIT_Sample" Assembly="MLIC" %>
<%@ Page language="c#" Codebehind="default.aspx.cs"
    Inherits="MSPress.MobWeb.MLICExample._default"
    AutoEventWireup="false" %>
<%@ Register TagPrefix="mobile"
Namespace="System.Web.UI.MobileControls"
    Assembly="System.Web.Mobile " %>

<mobile:Form id="Form1" runat="server" BackColor="PaleTurquoise">
    <mobile:Label id="l" runat="server"
        text="Multiline Text Input Sample"></mobile:Label>
    <cc1:MultiLineInput id="MultiLineInput1" runat="server"
        Cols="25" Rows="5" MaxLength="125"></cc1:MultiLineInput>
    <mobile:Label id="Result" runat="server"
        Text="Input text is: <empty>"></mobile:Label>
    <mobile:Command id="Command1" runat="server"
        Text="Get the text!"></mobile:Command>
</mobile:Form>
```

أما النص البرمجي في الخلفية فهو كالتالي:

```
using System;
using System.Web.UI.MobileControls;

namespace MSPress.MobWeb.MLICExample
{
    public class _default : System.Web.UI.MobileControls.MobilePage
    {
        protected MMIT_Sample.MultiLineInput MultiLineInput1;
        protected System.Web.UI.MobileControls.Label Result;
        protected System.Web.UI.MobileControls.Command Command1;

        override protected void OnInit(EventArgs e)
        {
            InitializeComponent();
            base.OnInit(e);
        }
```

```
        private void InitializeComponent()
        {
            this.Command1.Click +=
                new System.EventHandler(this.Command1_Click);
        }

        private void Command1_Click(object sender, System.EventArgs e)
        {
            Result.Text = "Input text is: "+ MultiLineInput1.Text;
        }
    }
}
```

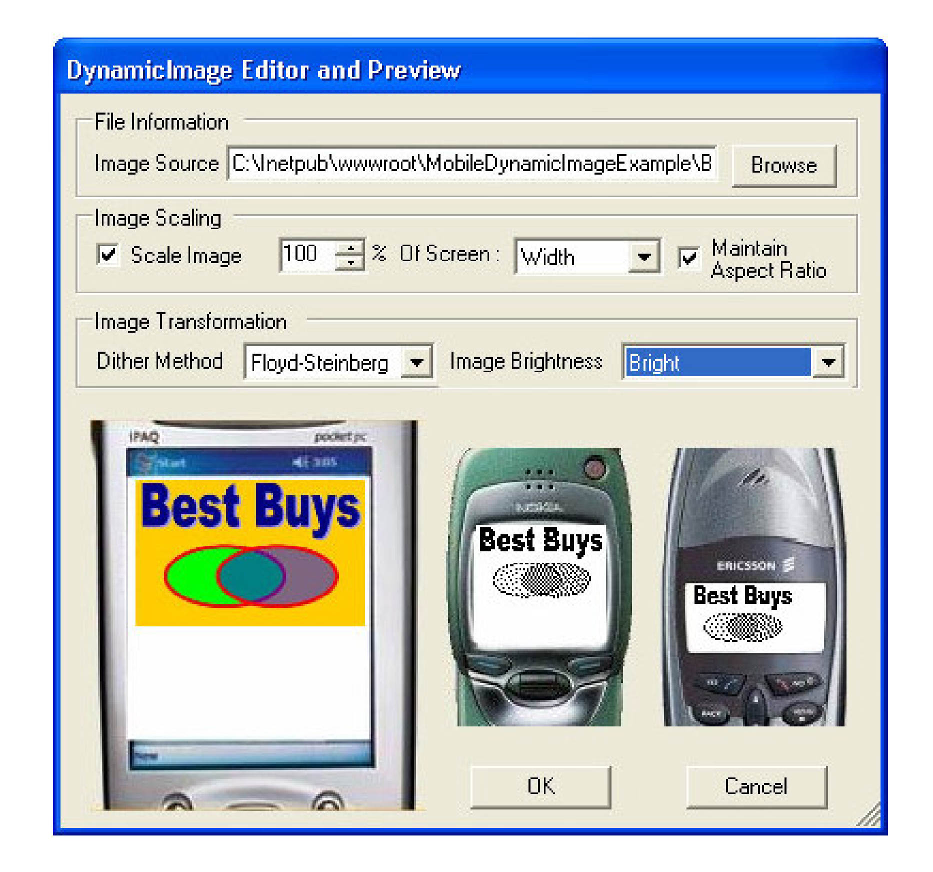
عنصر التحكم MobileDynamicImage

يحل عنصر التحكم هذا مشكلة تربك المطورين إذ يساعد في استخدام الصور ضمن التطبيق المحمول دون القلق من التجهيزات التي تشغله.

ذكرنا مسبقاً بأن أنواع المستعرضات المختلفة تدعم أنماط مختلفة من الملفات مثل JPG،GIF،WBMP، PNG... إلخ. في هذا الإطار يساعد عنصر التحكم MobileDynamicImage في أخذ صورة واحدة وتحويلها في زمن التشغيل إلى نمط الصورة الذي يدعمه الجهاز الزبون.

أهم المزايا التي يقدمها هذا العنصر التحكم هي:

- تحديد وتصحيح نمط الصورة المناسب وتحويلها بشكل مناسب
- دعم أنماط الصور GIF، JPEG، WBMP، PNG
- تغيير أبعاد الصورة تلقائياً ليناسب حجم شاشة العرض
- يقوم بعملية تخبئة للصورة المولدة بحيث لا يتم إنشاؤها أكثر من مرة
- يدعم تقنية تحويل الصورة الملونة إلى تدرجات الرمادي
- يوفر دعم كامل لمعاينة زمن التصميم ودعم كامل في Visual Studio.



إعداد التطبيق:

لا حاجة إلى إعدادات خاصة ضمن ملف Web.config لأن عنصر التحكم هذا لا يستخدم أي موائم خاص بالتجهيزات. ولكن يُفضل أن نحدد مجلد افتراضي خاص لوضع الصور. لذلك يمكن إضافة هذا النص إلى الملف Web.config

```
<appSettings>
    <add key="MobileDynamicImagePath"
value="c:\inetpub\wwwroot\dynimg\" />
    <add key="MobileDynamicImageURL" value="http://Myserver/dynimg/" />
</appSettings>
```

حيث يُعرِّف MobileDynamicImagePath المسار الفيزيائي للمجلد الخاص بالصور ويُعرِّف MobileDynamicImageURL المسار وفق ماهومُعرَّف في IIS.

تكون صيغة استخدام عنصر التحكم هذا على الشكل:

```
<%@ Register TagPrefix="mobDI" Namespace="MobileDynamicImage"
    Assembly=" MobileDynamicImage" %>

<mobDI:MobileDynamicImage
    runat="server"
    id="id"
    Alignment="{NotSet|Left|Centre|Right}"
    BackColor="backgroundColor"
    BreakAfter="{True|False}"
    Font-Bold="{NotSet|False|True}"
    Font-Italic="{NotSet|False|True}"
```

```
    Font-Name="fontName"
    Font-Size="{NotSet|Normal|Small|Large}"
    ForeColor="foregroundColor"
    StyleReference="StyleReference"
    Visible="{True|False}"
    Wrapping="{NotSet|Wrap|NoWrap}"

    AlternateText="AltText"
    ImageUrl="masterImageSource"
    NavigateUrl="targetURL"
    SoftkeyLabel="softkeyLabel"

    AutoConvert="{True|False}"
    AutoSizeImage="{True|False}"
    DynamicImageSource="string"
    ScalePercent="{0-100}"
    ScaleBasedOn="{ScreenWidth|ScreenHeight}"
    ImageBrightness=
        "{Auto|Very_Light|Light|Medium|Dark|Very_Dark}"
    ImageDitherMethod="{ThreshHold|Matrix|Floyd_Steinberg}"
    MaintainAspectRatio="{True|False}" >
</mobDI:MobileDynamicImage >
```

يوضح الجدول التالي خصائص عنصر التحكم MobileDynamicImage:

AutoConvert	True | False	في حال إسناد True إلى هذه القيمة ستتم عملية تحويل تلقائي للنمط المناسب وإلا سيتصرف عنصر التحكم هذا كعنصر تحكم Image عادي.
AutoSizeImage	True | False	إذا كانت قيمة هذه الخاصة True سيتم تعديل أبعاد الصورة بحسب الجهاز المستخدم اعتماداً على الخاصة ScaleBasedOn والخاصة ScalePercent.
DynamicImageSource	String	تعبر هذه الخاصة عن المسار إلى الصورة أوعنوان الصورة على الوب.
ScalePercent	Integer	قيمة في المجال من 0 إلى 100. في حال إسناد True إلى قيمة الخاصة AutoSizeImage سيتم تغيير أبعاد الصورة باستخدام هذه النسبة من الخاصة

		ScaleBasedOn
ScaleBasedOn	MobileDynamicImage. ScaleBasedOnType ScreenWidth|ScreenHeight	إذا تم إسناد True إلى القيمة AutoSizeImage سيتم تغيير حجم الصورة اعتماداً على هذه الخاصة إضافة إلى ScalePercent.
ImageBrightness	MobileDynamicImage. Brightness Auto|Very_Light|Light| Medium|Dark|Very_Dark	تحدد درجة الإضاءة للصورة المصدر.
ImageDitherMethod	MobileDynamicImage. DitherMethod ThreshHold|Matrix |Floyd_Steinberg	تحديد تقنية التداخل المستخدمة في تحويل الصور.
MaintainAspectRatio	True | False	إذا تم تعيين هذه الخاصة إلى True تتم عملية تغيير حجم الصورة مع المحافظة على نسبة الطول الأصلي إلى العرض الأصلي.

فيما يلي مثال بسيط يوضح استخدام عنصر التحكم هذا:

```
<%@ Register TagPrefix="mobile"
Namespace="System.Web.UI.MobileControls"
    Assembly="System.Web.Mobile" %>
<%@ Page language="c#"
    Inherits="System.Web.UI.MobileControls.MobilePage %>
<%@ Register TagPrefix="cc1" Namespace="MobileDynamicImage"
    Assembly="MobileDynamicImage" %>
<mobile:Form id="Form1" runat="server">
    <cc1:DynamicImage id="DynamicImage1" runat="server"
        ImageBrightness="Light" ImageDitherMethod="Floyd_Steinberg"
        DynamicImageSource=

"C:\Inetpub\wwwroot\MobileDynamicImageExample\BestBuys1.JPG">
    </cc1:DynamicImage>
</mobile:Form>
```


OWG1 - Openwave
File Info Edit Settings Location Help
Go http://content-lt56/MobileDynamicImage
OPENWAVE
Best Buys
Back


Pocket PC 2002
Emulator Help
Internet Explorer 4:53
http://content-lt56/mobiledynamici
Best Buys

القسم الحادي عشر:

الوصول إلى قواعد البيانات باستخدام التطبيقات المحمولة

الكلمات المفتاحية:

تطبيق، بيانات، اتصال، النص البرمجي، قاعدة بيانات، علائقية.

ملخص:

تحتاج التطبيقات المحمولة كغيرها من التطبيقات إلى الوصول إلى قواعد البيانات سواء لاستخدام تلــك البيانـــات للقــراءة أوالتحريـــر أو التعديل.

سنناقش في هذه الجلسة الأغراض المستخدمة لتأمين هذا الاتصال وتوافق كل غرض من هذه الأغــراض مـــع ســـيناريوهات الـــربط المختلفة.

أهداف تعليمية:

يتعرف الطالب في هذا الفصل على:

- مفهوم الوصول إلى البيانات في ASP.NET
- ربط البيانات التصريحي
- أغراض ADO.NET وطرق استخدامها
- مكونات البيانلات وطريقة استخدامها.

الوصول إلى البيانات

تتعامل أغلب تطبيقات ASP.NET مع مختلف أشكال البيانات، إذ يمكن ربط عناصر التحكم المحمولة بمصادر البيانات كما هي الحال في عناصر التحكم مثل SelectionList، وList، وObjectList.

يمكن أن تكون أغراض التعامل مع البيانات Collection أو ArrayList أو System.Data.DataSet، أو System.Data.DataTable.

تعتبر ADO.NET الجيل الأحدث من تقنيات الوصول إلى البيانات التي تم تطويرها من قبل شركة Microsoft، ولكن هذه التقنية لاتقدم بديلاً عن التقنية الأقدم المستعملة وهي ADO، إذ ما يزال بالإمكان استخدام هذا الغرض وإن كان يفتقد الكثير من الميزات، والأداء الذي تقدمه التقنية الجديدة.

استخدام ربط البيانات التصريحي في ASP.NET

احتوت العديد من الأمثلة التي قمنا باستعراضها في الجلسات الماضية، تعبيرات خاصة بالبيانات في صفحات نماذج الوب المحمولة تم تضمينها في التأشيرة <% #.... %>.

يعتبر هذا النمط مناسباً عند الرغبة بالوصول إلى عناصر البيانا. كذلك يمكن استخدام هذه الصيغة لتحقيق الأغراض التالية:

- تحديد مجموعات البيانات التي سيتم ربط عناصر التحكم إليها.
- استدعاء الطرق أوتقييم التعبيرات.

أمثلة على صيغة ربط البيانات التصريحي:

مصدر البيانات	مثال	الشرح
خاصة	`<%# TopTitle %>`	القيمة التي يتم إظهارها هي تلك المحددة للخاصة TopTitle ضمن النص البرمجي في الخلفية.
مجموعة	`<mobile:ObjectList id="ObjectList1" runat="server" LabelField="TeamName " DataSource = <%# MyArray %> />`	يتم إسناد الخاصة MyArray إلى الخاصة DataSource للغرض ObjectList. حيث تعبر MyArray عن مثل من نوع Collection أو ArrayList أو DataTable
تعبير	`<%# (TeamStats.Played + " Pts: " + TeamStats.Points) %>`	تتشكل القيمة الظاهرة هنا من تعبير يجمع خصائص الصف TeamStats ونص مباشر.

تنفيذ تابع	`<%# String.Format( "Position: {0}", TextBox1.Text. PadLeft(2,'0')) %>\`	يكون محتوى الصفحة في هذه الحالة هوخرج التابع String.Format
نتيجة طريقة	`<%# GetOdds(SelectionList1 .Selection.Text) %>`	GetOdds في هذا المثال هي طريقة في الصف خلف النص البرمجي. سيتم إدراج القيمة المعادة من تلك الطريقة ضمن صفحة الوب المحمولة.

يمكنك استخدام الربط التصريحي في أي مكان ضمن نموذج صفحة الوب المحمول مادام التعبير يعيد غرض صحيح.
يحتاج ربط البيانات إلى استدعاء الطريقة DataBind للعناصر المطلوب ربطها. يكفي استدعاء هذه الطريقة التابعة للغرض الممثل لصفحة الوب ضمن معالج حدث تحميل الصفحة، مما يؤدي لاستدعائها من أجل كل عناصر التحكم المحتواة ضمن الصفحة.

النص البرمجي التالي يوضح الصيغة الممكن استخدامها:

```
protected void Page_Load(Object sender, EventArgs e)
{
    this.DataBind();
}
```

قد يسبب استخدام هذه الطريقة أحياناً حدوث خطأ إذا كان التعبير المستخدم للربط يدل على غرض له القيمة null، فعلى سبيل المثال يمكن أن يظهر هذا الخطأ عند محاولة ربط قيمة الخاصة Selection لعنصر تحكم SelectionList قبل أن يقوم المستخدم باختيار أية قيمة للخاصة. لذا من الضروري في بعض الأحيان، تأجيل عملية الربط لحين التأكد من إسناد قيم للعناصر المطلوبة:

يوضح المثال التالي هذه الفكرة:

```
<%@ Page Inherits="MSPress.MobWeb.DeclDBEx.ExampleWebForm" Language="c#"
    CodeBehind="DeclarativeDataBinding.aspx.cs" AutoEventWireup="false" %>
<%@ Register TagPrefix="mobile" Namespace="System.Web.UI.MobileControls"
    Assembly="System.Web.Mobile" %>

<mobile:Form runat="server" id="Form1">
    <mobile:Label id="Label1" runat="server" StyleReference="title">
        <%# TopTitle %></mobile:Label>
    <mobile:ObjectList id="ObjectList1" runat="server"
        DefaultCommand="aSelection"
        LabelField="TeamName"
        DataSource = <%# MyArray %> >
        <Command Name="aSelection" Text="Show Details"/>
    </mobile:ObjectList>
</mobile:Form>

<mobile:Form runat="server" id="Form2">
    <mobile:Label id="Label2" runat="server" StyleReference="title">
        You selected <%# ObjectList1.Selection["TeamName"] %>
    </mobile:Label>
    <mobile:TextView id="txvDetail" runat="server">
```

```
    Played : <%# ObjectList1.Selection["Played"] %> <br>
    Points : <%# ObjectList1.Selection["Points"] %> <br>
    <%# String.Format("Position: {0}",
        ObjectList1.Selection["Position"].PadLeft(2,'0')) %>
    </mobile:TextView>
</mobile:Form>
```

ويكون الملف العامل في الخلفية هوالتالي:

```
using System;
using System.Collections;
using System.Web.UI.MobileControls;

namespace MSPress.MobWeb.DeclDBEx
{
    public class ExampleWebForm : MobilePage
    {
        protected System.Web.UI.MobileControls.Form Form1;
        protected System.Web.UI.MobileControls.Form Form2;
        protected System.Web.UI.MobileControls.ObjectList ObjectList1;
        private   ArrayList  _myArray;

        protected ArrayList MyArray
        {
            get { return _myArray; }
        }

        public string TopTitle
        {
            get { return "Season 2003 results"; }
        }

        override protected void OnInit(EventArgs e)
        {
            InitializeComponent();
            base.OnInit(e);
        }

        private void InitializeComponent()
        {
            this.Load += new System.EventHandler(this.Page_Load);
            this.ObjectList1.ItemCommand += new
                ObjectListCommandEventHandler(this.OnTeamSelection);
        }

        private void Page_Load(Object sender, EventArgs e)
        {
            if (!this.IsPostBack)
            {
                _myArray = new ArrayList();
                _myArray.Add(new TeamStats("Dunes",1,38,24,8,6,80));
                _myArray.Add(new TeamStats("Phoenix",2,38,20,10,8,70));
                _myArray.Add(new TeamStats("Eagles",3,38,20,9,9,69));
                _myArray.Add(new TeamStats("Zodiac",4,38,20,8,10,68));

                Form1.DataBind();
```

```
            }
        }

        private void OnTeamSelection(
            Object source,
            ObjectListCommandEventArgs args)
        {
            Form2.DataBind();
            this.ActiveForm = Form2;
        }
    }

    class TeamStats
    {
        private String  _teamName;
        private int _position, _played, _won, _drawn, _lost, _points;

        public TeamStats(String teamName,
            int position,
            int played,
            int won,
            int drawn,
            int lost,
            int points)
        {
            this._teamName = teamName;
            this._position = position;
            this._played = played;
            this._won = won;
            this._drawn = drawn;
            this._lost = lost;
            this._points = points;
        }

        public String TeamName { get { return this._teamName; } }
        public int    Position { get { return this._position; } }
        public int    Played   { get { return this._played; } }
        public int    Won      { get { return this._won; } }
        public int    Drawn    { get { return this._drawn; } }
        public int    Lost     { get { return this._lost; } }
        public int    Points   { get { return this._points; } }
    }
}
```

احتوت العديد من الأمثلة التي قمنا باستعراضها في الجلسات الماضية، تعبيرات خاصة بالبيانات في صفحات نماذج الوب المحمولة تم تضمينها في التأشيرة <% ....# %>.

يعتبر هذا النمط مناسباً عند الرغبة بالوصول إلى عناصر البيانا. كذلك يمكن استخدام هذه الصيغة لتحقيق الأغراض التالية:

- تحديد مجموعات البيانات التي سيتم ربط عناصر التحكم إليها.
- استدعاء الطرق أوتقييم التعبيرات.

استخدام ADO.NET

تستخدم جميع الأمثلة التي قدمناها والتي تتعلق بربط البيانات صفوف من النمط Collection مثل ArrayList. أما إذا كانت البيانات التي نريد الوصول إليها مخزنة في قاعدة بيانات فيجب علينا استخدام الصفوف الخاصة بـــ ADO.NET مثل صف DataSet و DataView.

يطرح الوصول إلى البيانات من جهة المخدم تحديات خاصة بسبب عدم تخزين صفحة الوب للحالة، بالإضافة إلى الحاجة المستمرة لتحديث البيانات والمترافق مع إنشاء أمثال مختلفة لصفحة الوب بهدف تخديم عدة زبائن.

توفر ADO.NET حلاً مناسباً لهذا النمط من المشاكل. إذ تقدم ADO.NET الصفوف DataSet و DataReader التي تعمل بصورة مستقلة عن مصدر البيانات.

فهم عمل أغراض ADO.NET:

DataSet:

تمثل أغراض DataSet البيانات الحقيقية التي يتعامل معها التطبيق. ولأن هذه العناصر مستقلة عن مصدر البيانات يمكنك تعديلها بصورة مستقلة. على أي حال يمكننا بسهولة إعادة نقل التعديلات إلى البيانات الأساسية.
تشبه البنية الداخلية لأغراض DataSet بنية قواعد البيانات العلائقية، فهي تحتوي على جداول، وأعمدة، وعلاقات، وقيود، ومناظير...
يمكن لأغراض DataSet أن تنتج عن استعلام قاعدة بيانات، كما يمكن إنشاء هذه الأغراض من ملفات XML. وبما أن غرض DataSet مستقل عن طبقة البيانات يمكننا العمل باستخدام نموذج برمجي موحد بغض النظر عن مصدر البيانات.

DataAdapter:

تتبنى صفوف DataAdapter الواجهة System.Data.IDataAdapter وهي مسؤولة عن تأهيل أغراض DataSet بالبيانات.
تعكس أغراض DataAdapter على أغراض DataSet، أية تعديلات تتم على البيانات في قاعدة البيانات.

Connection:

تتبنى هذه الأغراض الصف System.Data.IDbConnection وتمثل الاتصال الفيزيائي مع مصدر البيانات كمخدم MS SQL أوملف XML.

Command:

تعبر عن الصفوف التي تتبنى الواجهة System.Data.IDbCommand وتحتوي أوامر SQL المستخدمة للوصول إلى مصدر البيانات.

DataReader:

نلجأ أحياناً إلى استخدام أغراض DataReader كبديل عن استخدام DataSet وهي صفوف تتبنى الواجهة

System.Data.IDataReader وتوفر وصول فعال للقراءة فقط إلى مصدر البيانات. لا تقدم هذه الأغراض ميزات إجراء التعديلات التي تمنحها أغراض DataSet فهي مخصصة للقراءة فقط من مصدر البيانات.

اختيار مزود البيانات

تحدد الواجهات IDataAdapter، IDataReader، IDbConnection، IDbCommand كيفية وصول تطبيق إلى قاعدة البيانات.

مزود البيانات هومجموعة من الصفوف التي تتبنى الواجهات المذكورة وهي خاصة بقاعدة بيانات معينة.

في إطار العمل.Net 1.1 لدينا أربعة خيارات أساسية:

- مزود NET. SQL Server.
- مزود. OLEDB NET يستخدم للاتصال بقواعد البيانات باستخدام الواجهة OLEDB.
- مزود.NET ل Oracle بستخدم للاتصال بقواعد بيانات Oracle.
- مزود.NET الخاص بـــ ODBC.

يتبنى كل مزود أغراض Connection، وCommand، وDataAdapter، و DataReader فعلى سبيل المثال يتبنى SQL Server صفوف SQLConnection، SQLCommand، SQLDataAdapter.... الخ

لا بد لنا إذا لضمان عمل هذا الاتصال مع قواعد البيانات، استيراد فضاء الأسماء المناسب. ففي حالة MS SQL مثلاً نبدأ النص البرمجي بالصيغة:

```
using System.Data;
using System.Data.SqlClient;
```

أما في حالة مزود OLEDB فيصبح النص من الشكل :

```
using System.Data;
using System.Data.OleDb;
```

استخدام الغرض DataReader للقراءة المخصصة للبيانات

إذا كان الوصول الذي يتطلبه التطبيق لا يحتاج إلى إجراء تعديلات على البيانات، فإن استخدام الغرض DataReader يوفر بديل فعال عن استخدام الغرض DataSet.

لاستخدام هذا العنصر علينا تأسيس اتصال مع قاعدة البيانات، وتعريف أمر SQL لوضع البيانات ضمن غرض Command ثم

استخدام الطريقة ExecuteReader لهذا الغرض.

تعيد هذه الطريقة غرض DataReader يحتوي البيانات التي يمكن استخدامها كمصدر بيانات لعناصر التحكم المختلفة. ولا بد من الملاحظة هنا أن هذه العملية لا تتطلب أي غرض من النمط DataAdapter.

فيما يلي مثال بسيط يستخدم الغرض DataReader لتزويد عنصر تحكم List بالبيانات من قاعدة بيانات MS SQL هذا:

```
<%@ Register TagPrefix="mobile"
Namespace="System.Web.UI.MobileControls"
    Assembly="System.Web.Mobile" %>
<%@ Page language="c#" Codebehind="DataReaderExample.aspx.cs"
    Inherits="MSPress.MobWeb.DataRdrEx.DataReaderMobileWebForm" %>

<mobile:Form id="Form1" runat="server" Paginate="True">
    <mobile:List id="List1" runat="server"></mobile:List>
</mobile:Form>
```

ويكون النص البرمجي في الخلفية :

```
using System;
using System.Data;
using System.Data.SqlClient;
using System.Web.UI.MobileControls;

namespace MSPress.MobWeb.DataRdrEx
{
    /// <summary>
    /// Use the DataReader for efficient read-only access to data.
    /// </summary>
    public class DataReaderMobileWebForm
        : System.Web.UI.MobileControls.MobilePage
    {
        protected System.Web.UI.MobileControls.List List1;
        protected System.Web.UI.MobileControls.Form Form1;

        override protected void OnInit(EventArgs e)
        {
            InitializeComponent();
            base.OnInit(e);
        }
        private void InitializeComponent()
        {
            this.Load += new System.EventHandler(this.Page_Load);
        }

        private void Page_Load(object sender, System.EventArgs e)
        {
            // Use the DataReader to fetch a read-only dataset.
            String strConnectionString = "server=(local)\\NetSDK;" +
                "database=pubs;Trusted_Connection=yes";
```

```
            SqlConnection myConnection =
                new SqlConnection(strConnectionString);
            SqlCommand myCommand =
                new SqlCommand("select * from Authors", myConnection);

            myConnection.Open();

            SqlDataReader dr = myCommand.ExecuteReader();

            List1.DataSource = dr;
            List1.DataTextField="au_lname";
            List1.DataBind();

            myConnection.Close();
        }
    }
}
```

أما خرج هذا البرنامج فيكون على الشكل :



استخدام الغرض DataSet في عملية الربط بالبيانات

في الكثير من التطبيقات يقدم الغرض DataReader جميع الوظائف المطلوبة ولكن يصبح استخدام هذا الغرض غير كافي في التطبيقات التي تستلزم مناقلات طويلة أو عمليات تحديث على البيانات ضمن قاعدة البيانات.

يقدم الغرض DataSet العديد من الفوائد أهمها احتواؤه على معلومات حول القيود المُعرَّفة ضمن قاعدة البيانات. لذلك يمكنك إجراء عمليات التعديل على البيانات مع إمكانية التنبه إلى أي تجاوز لأحد القيود عند تحديث البيانات ضمن غرض DataSet قبل تعديلها فعلياً ضمن قاعدة البيانات، مما يؤدي لضمان انسجام التعديلات مع القيود المعرفة في قاعدة البيانات عند تطبيق التحديثات على القاعدة.

تشبه عملية الوصول إلى قاعدة البيانات لتأهيل غرض DataSet، العملية التي استخدمناها مع الغرض DataReader. إذ نقوم بتعريف أمر SQL لجلب البيانات من قاعدة البيانات باستخدام غرض DataAdapter بدلاً عن غرض Command.

يوضح المثال التالي كيفية استخدام الغرض DataSet لتأهيل عنصر التحكم ObjectList1:

```
// Use the DataAdapter to fill a dataset.
String strConnectionString =
    "server=(local)\\NetSDK;database=pubs;Trusted_Connection=yes";
SqlConnection myConnection =
    new SqlConnection(strConnectionString);
SqlDataAdapter myCommand =
    new SqlDataAdapter("select * from Authors", myConnection);

DataSet ds = new DataSet();
myCommand.Fill(ds, "Authors");
ObjectList1.DataSource = ds.Tables["Authors"].DefaultView;
ObjectList1.LabelField = "au_lname";
ObjectList1.AutoGenerateFields = true;
ObjectList1.DataBind();
```

نلاحظ بأن الغرض DataSet يحتوي أغراض DataTables التي تحتوي بدورها أغراض DataRow و DataColumn. تقدم هذه الصفوف معاً مجال واسع من الوظائف للتعامل مع البيانات.

إنشاء تطبيق وب محمول يقوم بتحديث البيانات

في حال كان التطبيق المراد إنشاؤه يحتاج إلى إجراء تعديلات على معطيات قاعدة البيانات، تتلخص الطريقة الأفضل في إجراء التعديلات على غرض DataSet وتخزينها ثم تطبيق هذه التعديلات على قاعدة البيانات باستخدام الغرض DataAdapter.

أما إذا قام المستخدم بإجراء تعديلات من شأنها أن تؤثر على سجل وحيد، فنستطيع تبنّي الطريقة الواردة في المثال التالي:

```
<%@ Page language="c#" Codebehind="DataUpdateExample.aspx.cs"
    Inherits="MSPress.MobWeb.DataUpdateEx.DataUpdateMobileWebForm" %>
<%@ Register TagPrefix="mobile"
    Namespace="System.Web.UI.MobileControls"
    Assembly="System.Web.Mobile" %>

<mobile:Form id="Form1" runat="server" Paginate="True">
    <mobile:ObjectList id="ObjectList1" runat="server">
```

```
        <Command Name="EditCommand" Text="Edit Details"/>
    </mobile:ObjectList>
</mobile:Form>

<mobile:Form id="Form2" runat="server">
    <mobile:Label id="Label1" runat="server"
        text="Edit Author Details" StyleReference="title"/>
    <mobile:Label runat="server">
        Author ID: <%# ObjectList1.Selection["au_id"] %>
    </mobile:Label>
    First Name:
    <mobile:TextBox id="TextBox1" runat="server" MaxLength="20"
        Text='<%# ObjectList1.Selection["au_fname"]%>' />
    Last Name:
    <mobile:TextBox id="TextBox2" runat="server" MaxLength="40"
        Text='<%# ObjectList1.Selection["au_lname"]%>' />
    <mobile:Label id=Label3 runat="server"
        StyleReference="error" Visible="false"/>
    <mobile:Command id="Command1" runat="server" Text="Save"
        CommandName="Save" />
    <mobile:Command id="Command2" runat="server" Text="Cancel"
        CommandName="Cancel" />
</mobile:Form>
```

ويكون النص البرمجي في الخلفية:

```
using System;
using System.Data;
using System.Data.SqlClient;
using System.Web.UI.MobileControls;
using System.Web.UI.WebControls;

namespace MSPress.MobWeb.DataUpdateEx
{
    /// <summary>
    /// Use the DataReader to fetch the data.
    /// </summary>
    public class DataUpdateMobileWebForm
        : System.Web.UI.MobileControls.MobilePage
    {
        SqlConnection myConnection;

        protected System.Web.UI.MobileControls.ObjectList ObjectList1;
        protected System.Web.UI.MobileControls.Form       Form1;
        protected System.Web.UI.MobileControls.Form       Form2;
        protected System.Web.UI.MobileControls.Label      Label3;
        protected System.Web.UI.MobileControls.Command    Command1;
        protected System.Web.UI.MobileControls.Command    Command2;
        protected System.Web.UI.MobileControls.TextBox    TextBox1;
        protected System.Web.UI.MobileControls.TextBox    TextBox2;

        override protected void OnInit(EventArgs e)
        {
```

```
            InitializeComponent();
            base.OnInit(e);
        }
        private void InitializeComponent()
        {
            this.Load += new System.EventHandler(this.Page_Load);
            this.ObjectList1.ItemCommand += new
                ObjectListCommandEventHandler(this.OnEditCommand);
            this.Command1.ItemCommand += new
                CommandEventHandler(this.CancelConfirmEdit);
            this.Command2.ItemCommand += new
                CommandEventHandler(this.CancelConfirmEdit);
        }

        private void Page_Load(object sender, System.EventArgs e)
        {
            // Use the DataReader to fetch a read-only data set.
            String strConnectionString = "server=(local)\\NetSDK;" +
                "database=pubs;Trusted_Connection=yes";
            myConnection = new SqlConnection(strConnectionString);

            if (!IsPostBack) BindList();
        }

        private void BindList()
        {
            SqlCommand myCommand =
                new SqlCommand("select * from Authors", myConnection);
            myConnection.Open();
            SqlDataReader dr = myCommand.ExecuteReader();

            ObjectList1.DataSource = dr;
            ObjectList1.LabelField = "au_lname";
            ObjectList1.AutoGenerateFields = true;
            ObjectList1.DataBind();

            // The field names of au_id, au_lname, and au_fname
            // do not provide good titles, so change them in the
            // AllFields collection.

ObjectList1.AllFields[ObjectList1.AllFields.IndexOf("au_id")]
               .Title = "Author ID";

ObjectList1.AllFields[ObjectList1.AllFields.IndexOf("au_fname")]
               .Title = "First Name";

ObjectList1.AllFields[ObjectList1.AllFields.IndexOf("au_lname")]
               .Title = "Last Name";
        }

        /// <summary>
        /// Called when the user clicks the 'Edit Details' link
```

```
        /// </summary>
        protected void OnEditCommand(
            Object source,
            ObjectListCommandEventArgs args)
        {
            // DataBind the form to insert the selected item details.
            Form2.DataBind();
            this.ActiveForm = Form2;

            Label3.Visible = false;
            Command1.Visible = true;
            Command2.Visible = true;
            Command2.Text = "Cancel";
        }

        /// <summary>
        /// Called when a user clicks on either 'Save' or 'Cancel'
button
        /// on Edit screen
        /// </summary>
        private void CancelConfirmEdit(Object sender, CommandEventArgs
e)
        {
            if (e.CommandName == "Save") {
                SaveChanges();
            }
            else
            {
                // Go back to the List View.
                this.ActiveForm = Form1;
                ObjectList1.ViewMode = ObjectListViewMode.List;
            }

            BindList();
        }

        private void SaveChanges()
        {
            String updateCmd = "UPDATE Authors SET au_lname = @LName, "
+
                "au_fname = @FName where au_id = @Id";

            SqlCommand myCommand = new SqlCommand(updateCmd,
myConnection);

            myCommand.Parameters.Add(
                new SqlParameter("@Id", SqlDbType.NVarChar, 11));
            myCommand.Parameters.Add(
                new SqlParameter("@LName", SqlDbType.NVarChar, 40));
            myCommand.Parameters.Add(
                new SqlParameter("@FName", SqlDbType.NVarChar, 20));
```

```
            myCommand.Parameters["@Id"].Value =
                ObjectList1.Selection["au_id"];
            myCommand.Parameters["@LName"].Value = TextBox2.Text;
            myCommand.Parameters["@FName"].Value = TextBox1.Text;

            myCommand.Connection.Open();

            try
            {
                myCommand.ExecuteNonQuery();
                Label3.Text = "Record Updated";
            }
            catch (SqlException)
            {
                Label3.Text = "ERROR: Could not update record";
            }

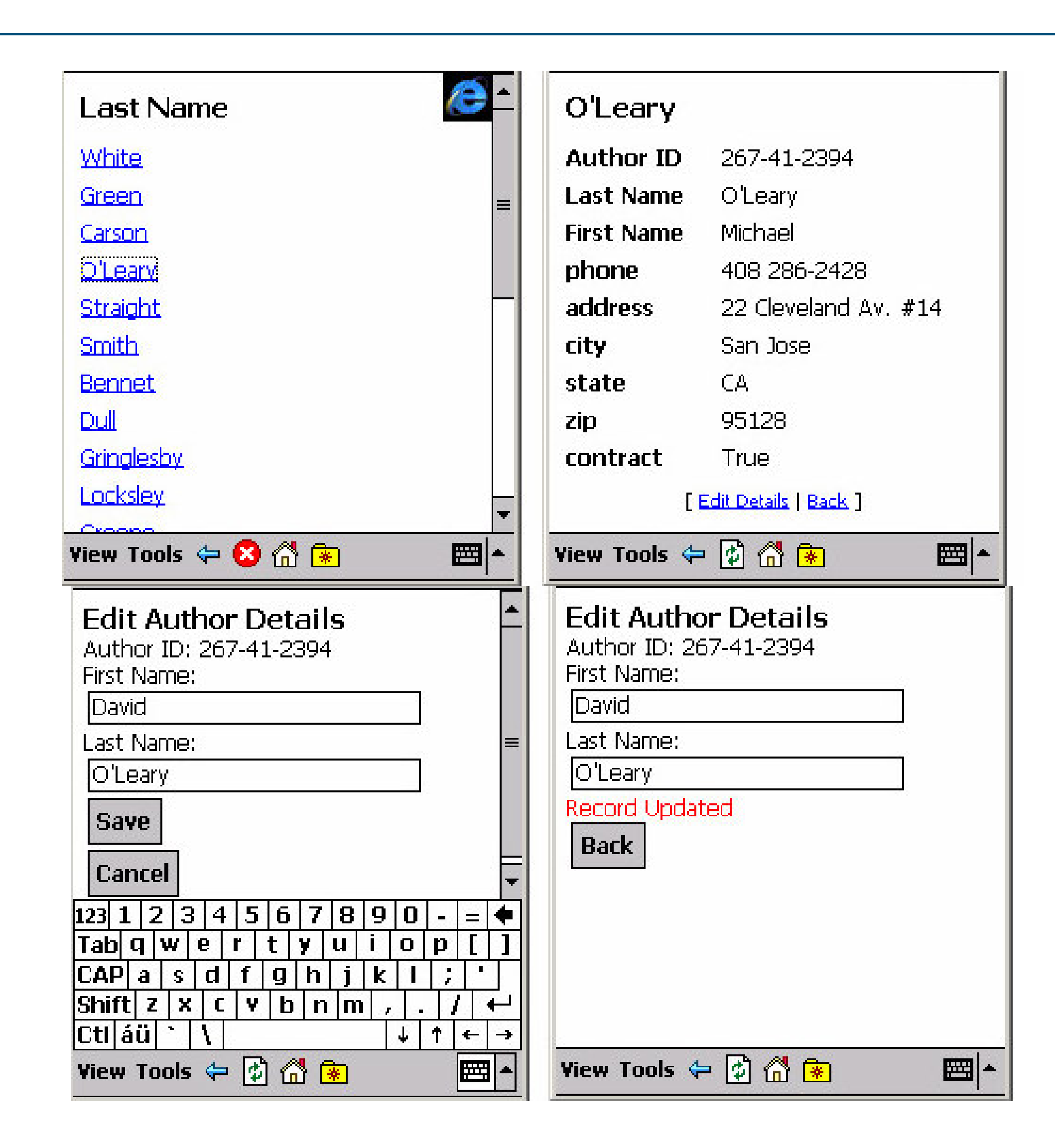
            myCommand.Connection.Close();

            Label3.Visible = true;
            Command1.Visible = false;
            Command2.Visible = true;
            Command2.Text = "Back";
        }
    }
}
```

يستخدم التطبيق في هذا المثال عنصر التحكم ObjectList لإظهار البيانات المخزنة ضمن الجدول authors في قاعدة البيانات Pubs.

يُعرِّف النص البرمجي، العنصر <Command> ضمن ملف aspx بحيث يساعد المستخدم في تحرير التفاصيل. فعند الضغط على هذا العنصر يظهر النموذج Form2 مبيناً قيم الحقول الحالية باستخدام عناصر تحكم مثل TextBox تساعد في عملية التحرير.

يبين الشكل التالي شكل الواجهة عند تنفيذ هذا البرنامج :



يُستخدم النموذج المستخدم لتحرير الاسم والشهرة، زرين: يكون الأول مخصص لتثبيت التعديلات ويكون الثاني مخصص لتجاوزها. وتجري عملية معالجة الضغط على هذين الزرين باستخدام طريقة واحدة لمعالج الحدث OnItemCommand هي CancelConfirmEdit.

يتوضع النص البرمجي المسؤول عن عملية التعديل ضمن الطريقة SaveChanges. نلاحظ أننا قمنا هنا أيضاً باستخدام معاملات SQL المسبوقة بإشارة (@)

```
String updateCmd = "UPDATE Authors SET au_lname = @LName, " +
                   "au_fname = @FName where au_id = @Id";
SqlCommand myCommand = new SqlCommand(updateCmd, myConnection);
```

يقوم النص البرمجي أيضاً بإضافة أغراض معاملات SQL وهي SQLParameter إلى غرض SQLCommand مع تحديد نمط البيانات لكل منها وذلك بالتعبير:

```
myCommand.Parameters.Add(new SqlParameter("@Id", SqlDbType.NVarChar,
11));
```

يحدد النص البرمجي بعدها القيم المطلوب استخدامها لهذه المعاملات وذلك بالصيغة :

```
myCommand.Parameters["@LName"].Value = txtLName.Text;
```

يمكننا بعد ذلك تنفيذ الاستعلام بطريقتين: الأولى هي باستخدام الطريقة Execute وفي هذه الحالة تجري عملية إعادة غرض DataSet، والثانية هي باستخدام الطريقة ExecuteNonQuery وفي هذه الحالة يجري تنفيذ الأمر دون إعادة غرض DataSet وهذا ما نفذناه باستخدام الصيغة:

```
myCommand.ExecuteNonQuery();
```

نستطيع تطبيق عمليات الحذف والإدراج بطريقة مشابهة وذلك بتغيير نص استعلام SQL.

بناء مكونات البيانات باستخدام بيئة Visual Studio

تقدم بيئة Visual Studio العديد من الأدوات التي تساعد المطور للعمل على البيانات.

يساعد مصمم DataSet Designer في التعامل مع أغراض قواعد البيانات باستخدام مخطط قاعدة البيانات، حيث يمكن إنشاء قاعدة بيانات تحتوي جداول، وصفوف، وأعمدة، ومفاتيح، وأدلة، وعلاقات، وقيود. يجري تقديم كل ذلك بطريقة مرئية تفاعلية.

سنتعرف في ما تبقى من هذه الجلسة على أداتين مهمتين أيضاً هما Server Explorer و Component Designer.

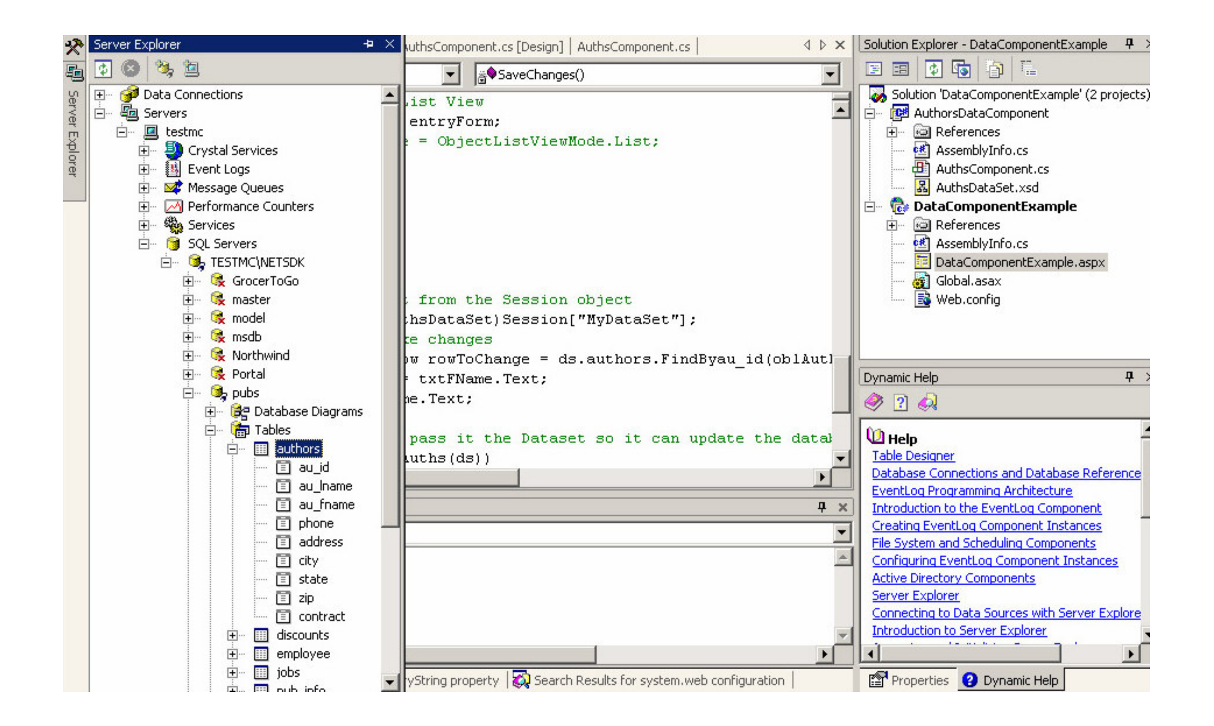
استخدام مستكشف المخدم Server Explorer:

تسمح هذه الأداة بالوصول والتجول ضمن مصادر البيانات المختلفة المتوفرة لتطبيقك. يمكنك الوصول إلى هذه الأداة باستخدام الخيار Server Explorer من القائمة View.

يمكنك باستخدام هذه الأداة:

- تأسيس اتصال مع مخدم قاعدة بيانات SQL وقواعد البيانات الأخرى.
- الاتصال مع مخدمات SQL لإظهار البيانات التي تحتويها.
- الاتصال بالأنظمة على شبكتك وإظهار الخدمات التي تقدمها بما فيها قواعد البيانات، وسجلات الأحداث، وأرتال الرسائل، وعددات الأداء.
- إظهار المعلومات المتعلقة بخدمات الوب المتوفرة ومخطط الطرق التي توفرها.

توضح اللقطة التالية عمل هذه الأداة:



بناء مكونات البيانات باستخدام بيئة Visual Studio

يمكن للتطبيقات البسيطة الوصول إلى قواعد البيانات مباشرة بالشكل الذي قمنا بتوضيحه في الجزء السابق.

على أي حال، من المفيد عندما نتكلم عن تطبيقات أكثر تعقيداً استخدام نموذج متعدد الطبقات (n-Tier) حيث يتم فصل الطبقات التي تقوم بمعالجة واجهة المستخدم عن تلك التي تهتم بمعالجة البيانات. بحيث يتم خلق مكونات مهمتها التعامل مع البيانات بالنيابة عن تلك الطبقات.

في بيئة متعددة الطبقات تقوم صفوف واجهة المستخدم باستدعاء مكونات البيانات التي تقوم بتطبيق منطق العمل. وتستدعي مكونات الطبقة الوسيطة مكونات أخرى مهمتها استعادة البيانات وتحديثها.

في مثل هذه البيئة، تعد أغراض DataSet مثالية لتأمين عملية نقل البيانات بين المكونات كونها منفصلة تماماً عن قاعدة البيانات مع قدرتها كما ذكرنا على الاحتفاظ بمعلومات عن العلاقات والقيود، تقوم بتطبيقها عند تعديل البيانات.

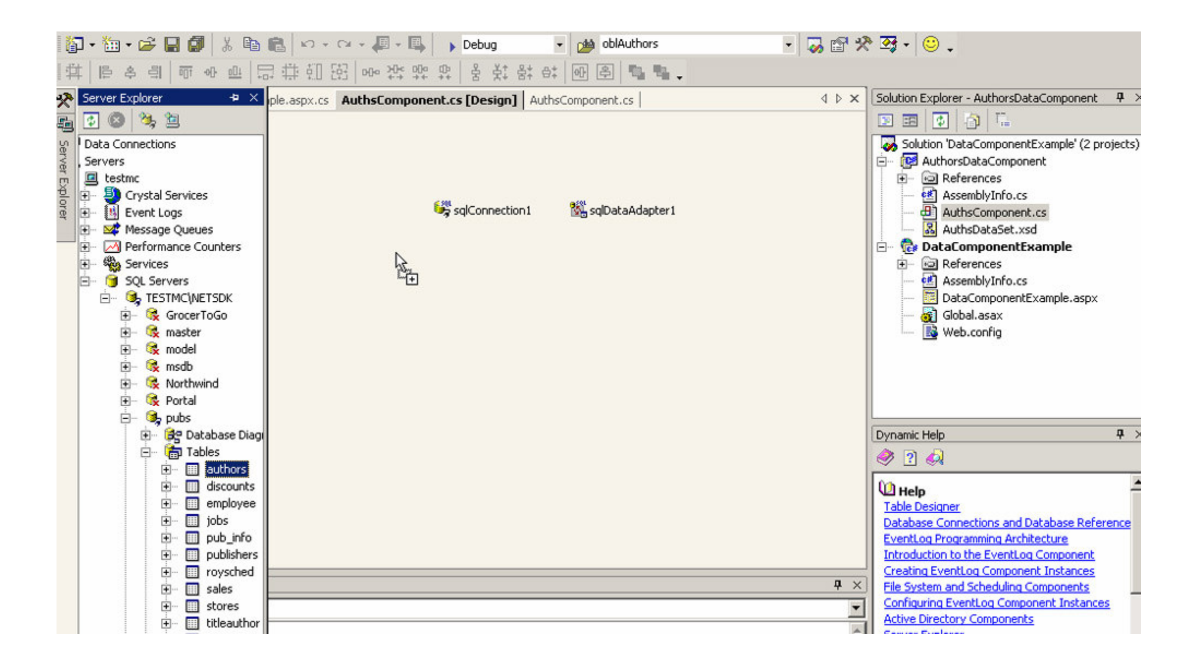
توفر بيئة Visual Studio.NET أداة لتصميم المكونات مرافقة لأداة Server Explorer مما يجعل من عملية بناء مكونات بيانات عملية سهلة.

لتوضيح عمل هذه الأدوات سنقوم بتعديل المثال الوارد في الشريحة السابقة باستخدام مكونات البيانات:

- سنقوم في البداية بإنشاء مشروع جديد من نمط مكتبة صفوف C# ونسميه AuthorsDataComponent. سيتم بهذه العملية إنشاء مشروع يتم تفسيره إلى لغة التجميع وهويحتوي بصورة داخلية صف بالاسم Class1.cs سنقوم بحذفه لأننا

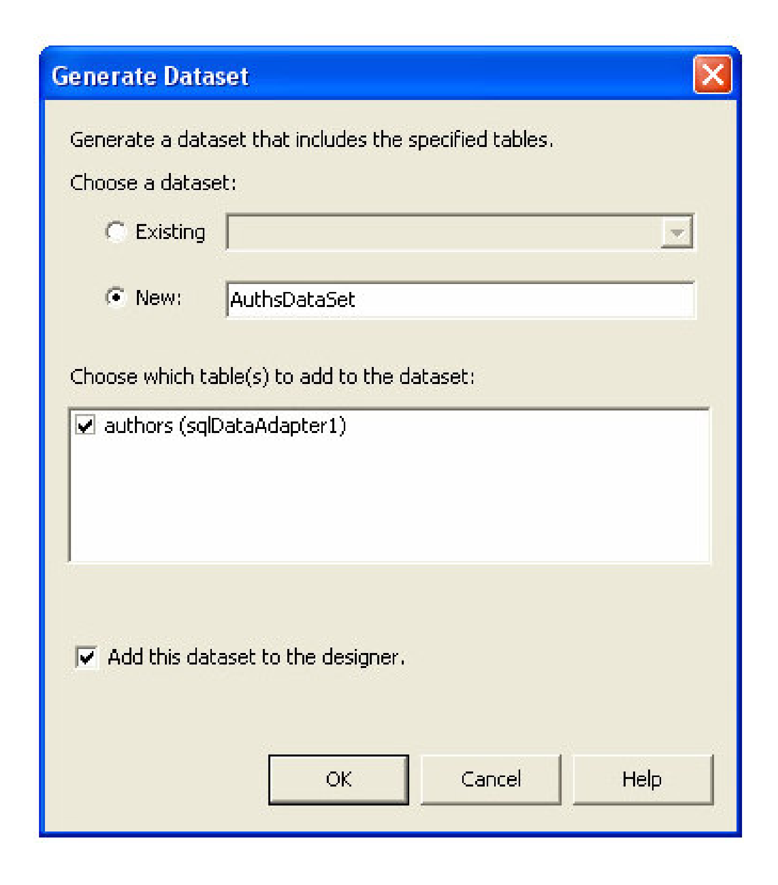
سنقوم بإنشاء صف آخر باستخدام مصمم المكونات.

- سنقوم بالنقر بالزر الأيمن على ملف المشروع واختيار خيار Add من قائمة context ثم نختار Add Component.
- في نافذة NewItem نختار Component Class ثم ندُخل اسم الملف AuthsComponent.cs.
- يقوم عندها Visual Studio بإضافة AuthsComponent.cs إلى مشروعنا ويقوم بفتحه ضمن وضعية التصميم.
- نقوم الآن بالنقر على مستكشف المخدم لمحاولة الوصول إلى الجدول authors ضمن قاعدة البيانات pubs.
- نقوم بعدها بعملية جر الجدول authors إلى مساحة التصميم بحيث يتم تلقائياً إنشاء غرض SQLConnection وغرض SQLDataAdapter ويتم تلقائياً توليد النص البرمجي المرتبط بتلك الأغراض المضافة متضمناً عبارة الاتصال الصحيحة مع غرض SQLCommand لإجراء علميات الاختيار، والإضافة، والتعديل والحذف.



يقوم المكون الذي قمنا بإنشاءه بالاتصال مع الصفوف الأخرى عن طريق إرسال واستقبال أغراض DataSet.

يساعد مصمم المكونات في توليد غرض DataSet خاص بالجدول الذي قمنا باختياره. نختار القائمة Data ثم الخيار Generate DataSet وفي هذه النافذة يجب علينا النقر على New ثم ادخال اسم AuthsDataSet كما هوموضح في الشكل التالي:



عند النقر على OK، يجري إنشاء ملف XML خاص بغرض DataSet باسم AuthsDataSet.xsd وتجري إضافته إلى المشروع ثم يجري إنشاء مثيل من هذا الغرض الجديد باسم authsDataSet1 وإضافته إلى شاشة التصميم.

يحتوي مكون البيانات المنشأ على جميع الإمكانيات اللازمة للتعامل مع البيانات ضمن الجدول authors.

لايقدم النص البرمجي الذي تم توليده لهذا الصف أي شيء لتطبيقنا ما لم تجر إضافة طريقة لتأهيل غرض DataSet وتجعله متوفراً للعالم الخارجي. إذاً يجب إضافة الخاصة العامة التالية إلى الصف الذي يقوم بتغذية عناصر الصف authsDataSet1 كما يلي:

```
/// <summary>
/// Returns a dataset of all authors in the authors table of the pubs
database
/// </summary>
public AuthsDataSet AllAuthors
{
    get
    {
        // Update class member dataset.
        this.sqlDataAdapter1.Fill(this.authsDataSet, "authors");
        return this.authsDataSet1;
    }
```

تقوم الطريقة Fill الخاصة بـ SqlDataAdapter بفتح اتصال بصورة أوتوماتيكية مع قاعدة البيانات وقراءة البيانات إلى غرض DataSet ثم إغلاق الاتصال.

كذلك يجب على هذا المكون تبني الطريقة الخاصة بتعديل قاعدة البيانات عند أي تغيير على البيانات. يجعل النص البرمجي الذي قام مصمم المكونات بتوليده من عملية التحديث غاية في السهولة.

قام مصمم المكونات بإعداد غرضٍ SQLDataAdapter مع غرض SqlCommand المناسب لعملية الإدراج، الحذف...
عند استدعاء طريقة Update الخاصة بغرض SQLDataAdapter يتم تمرير غرض DataSet إلى هذا الغرض محتوياً التعديلات التي تمت.
يقوم المحرك بعدها بتطبيق هذه التعديلات لكل صف من الصفوف ضمن غرض DataSet الذي تمت عليه عملية الإضافة أوالحذف أوالتحديث.

يكون النص البرمجي الواجب إضافته لتبني طريقة عامة لتحديث قاعدة البيانات كما يلي:

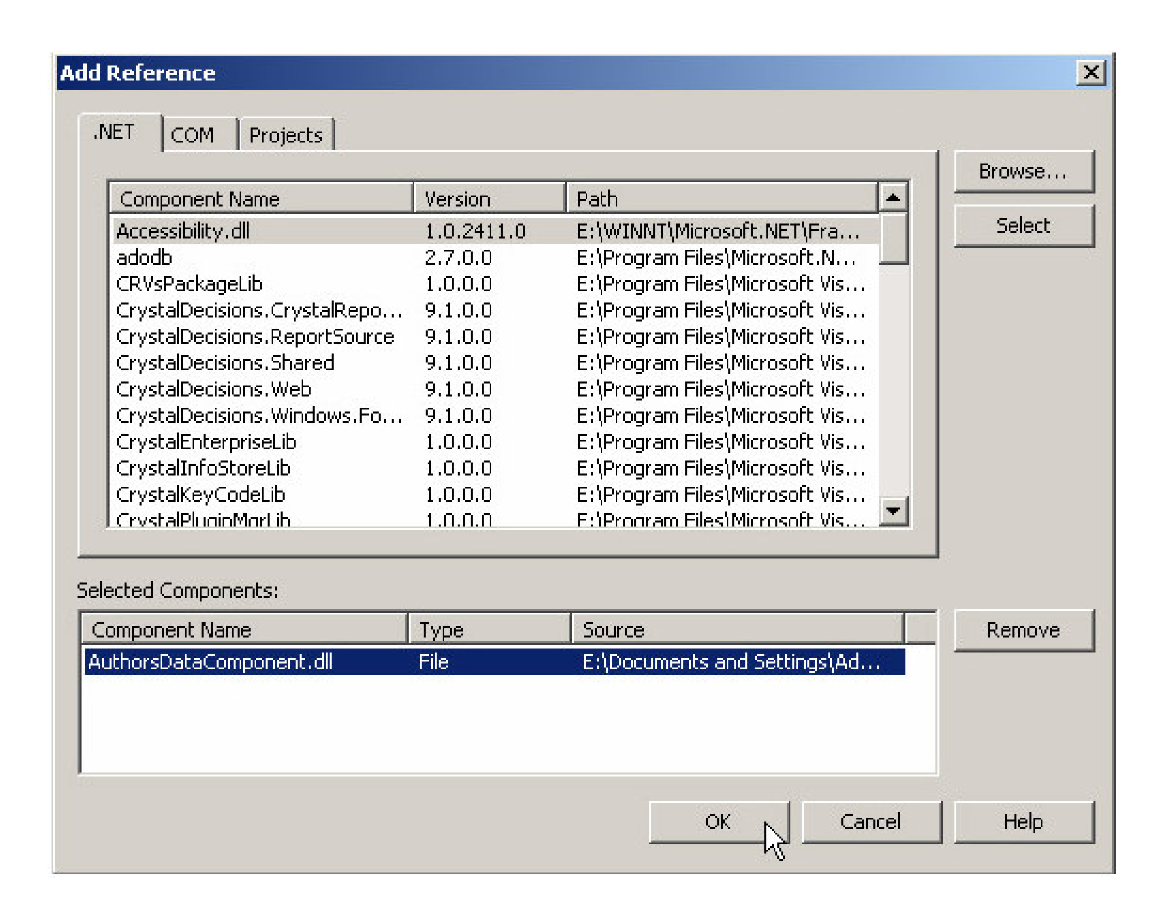
```
/// <summary>
/// Take a DataSet, including changes, and apply it to the database.
/// </summary>
public bool UpdateAuths(AuthsDataSet DataChanges)
{
    bool boolRetval;
    try
    {
        this.sqlDataAdapter1.Update(DataChanges, "authors");
        boolRetval = true;
    }
    catch(Exception)
    {
        boolRetval = false;
    }
    return boolRetval;
}
```

استخدام مكون البيانات ضمن تطبيق الوب

عند إزالة النص البرمجي الخاص بمعالجة البيانات من صف التطبيق الرئيسي يصبح أكثر ترتيباً مما يساعد في التركيز على تقديم واجهة أكثر مناسبة للمستخدم.

سنقوم بإعادة صياغة بعض الأجزاء من المثال الذي رأيناه في تحديث قواعد البيانات في تطبيقات الوب المحمولة لنطبق عملية استخدام مكونات البيانات.

لاستخدام مكون البيانات يجب إضافة مرجع إلى هذا المكون ضمن المشروع وذلك من خلال نافذة Add Reference:



يتوجب علينا بعدها إضافة تصريح عن هذا المكون في بداية النص البرمجي في الخلفية وذلك بالصيغة:

```
using MSPress.Mob.Web.AuthorsDataComponent;
```

تتلخص المرحلة التالية في إنشاء مكون البيانات كعنصر خاص ضمن الصف وذلك بالصيغة:

```
private AuthsComponent myDataComp;
```

```
private void Page_Load(object sender, System.EventArgs e)
{
// Create the data component each time the application
// returns to the server.
myDataComp = new AuthsComponent();

if (!IsPostBack)
    BindList();
}
```

ويكون النص البرمجي للطريقة BindList كما يلي:

```
public void BindList()
{
    // Use the DataComponent to fetch a dataset.
```

```
    AuthsDataSet ds = myDataComp.AllAuthors;

    ObjectList1.DataSource = ds.Tables["authors"].DefaultView;
    ObjectList1.LabelField = "au_lname";
    ObjectList1.AutoGenerateFields = true;
    ObjectList1.DataBind();

    // The field names of au_id, au_lname, and au_fname do not provide
    // good titles, so change them in the AllFields collection.
    ObjectList1.AllFields[ObjectList1.AllFields.IndexOf("au_id")].Title
        = "Author ID";

ObjectList1.AllFields[ObjectList1.AllFields.IndexOf("au_fname")].Title
        = "First Name";

ObjectList1.AllFields[ObjectList1.AllFields.IndexOf("au_lname")].Title
        = "Last Name";

    // Store the DataSource in a session variable so that
    // it can persist across multiple postbacks.
    Session["MyDataSet"] = ds;
}
```

تصبح الطريقة SaveChanges بعد إعادة كتابتها لاستخدام مكون البيانات على الشكل:

```
private void SaveChanges()
{
    // Retrieve the dataset from the Session object.
    AuthsDataSet ds = (AuthsDataSet)Session["MyDataSet"];
    // Find the row and make changes.
    AuthsDataSet.authorsRow rowToChange =
        ds.authors.FindByau_id(ObjectList1.Selection["au_id"]);
    rowToChange.au_fname = TextBox1.Text;
    rowToChange.au_lname = TextBox2.Text;

    // Call the UpdateAuths method of data component.
    // Pass it the dataset so that it can update the database.
    if (myDataComp.UpdateAuths(ds))
        Label3.Text = "Record Updated";
    else
        Label3.Text = "ERROR: Could not update record";

    Label3.Visible = true;
    Command1.Visible = false;
    Command2.Visible = true;
    Command2.Text = "Back";
}
```

عند إزالة النص البرمجي الخاص بمعالجة البيانات من صف التطبيق الرئيسي يصبح أكثر ترتيباً مما يساعد في التركيز على تقديم واجهة أكثر مناسبة للمستخدم.

سنقوم بإعادة صياغة بعض الأجزاء من المثال الذي رأيناه في تحديث قواعد البيانات في تطبيقات الوب المحمولة لنطبق عملية استخدام مكونات البيانات.

لاستخدام مكون البيانات يجب إضافة مرجع إلى هذا المكون ضمن المشروع وذلك من خلال نافذة Add Reference:

تزود بيئة Visual Studio.NET مصمم مكونات مرافق لأداة Server Explorer مما يجعل من عملية بناء مكونات بيانات عملية سهلة.

- لتوضيح عمل هذه الأدوات سنقوم بتعديل المثال الوارد في الشريحة السابقة باستخدام مكونات البيانات عوضاً عن استخدام منطق معالج البيانات المبيت.

القسم الثاني عشر:

الموضوع الأول: إدارة الحالة

الكلمات المفتاحية:

تطبيق، حالة، جلسة، كعكة، غرض

ملخص:

لما كان بروتوكول HTTP هو بروتوكول لا يحافظ على الحالة بين طلبات HTTP المتتالية، فلا بد من إيجاد طريقة لإدارة الحالة وهذا ما سنقوم بتغطيته في هذه الجلسة.

أهداف تعليمية:

يتعرف الطالب في هذا الفصل على:

- مفهوم الحالة وإدارتها.
- حالة الجلسة وطرق حفظها.
- حالة التطبيق.

إدارة الحالة

سنحتاج غالباً عند بناء تطبيقات الوب الديناميكية إلى آلية لتخزين المعلومات بين طلبات الزبون. لسوء الحظ لايحفظ البروتوكول HTTP معلومات الحالة بمعنى أنه يجب اللجوء إلى طريقة أخرى لتحقيق هذا الغرض.

كان المطورون في النسخة السابقة لـ ASP.NET وهي ASP يستخدمون طرق مختلفة منها الكعكات لتتبع المستخدمين والتعرف عليهم باستخدام ما يسمى SESSION_ID لاستعادة المعلومات المخزنة على المخدم والخاصة بهذا المستخدم.
لكن هذه الحلول تُعتبر غير مناسبة في أغلب التجهيزات المحمولة كون المستعرضات في هذه التجهيزات لاتدعم الكعكات.

إلا أن ASP.NET وفرت مجموعة من الآليات المختلفة للحفاظ على الحالة بعضها استمرار لما تم استخدامه في ASP وبعضها جديد. هذه الآليات هي التالية:

- **حالة الجلسة** : تسمح بالمحافظة على المتحولات والأغراض ضمن غرض Session في حال وجود عدة طلبات واستجابات.
- **المتحولات المخفية:** تسمح هذه المتحولات بإدامة قيم المتحولات والأغراض عن طريق إعادة إرسال البيانات للزبون من خلال حقول مخفية.
- **ViewState:** تسمح بالحفاظ على القيم الخاصة بنموذج وب محمول على المخدم. يقوم محرك زمن التشغيل بتخزين المعلومات ضمن مثل من الصف System.Web.UI.StateBag والذي يتم تخزينه ضمن غرض Session.
- **حالة التطبيق:** تساعد في الحفاظ على المتغيرات والأغراض الخاصة بتطبيق عبر عدة طلبات من عدة زبائن.

تتطلب الطرق الثلاث الأولى من المخدم التعرف على زبون ليتمكن من متابعة طلباته المتتالية مما يستلزم إرسال مُعرِّف وحيد إلى الزبون مع كل استجابة، يقوم الزبون بدوره بإرساله مع الطلب التالي. في المستعرضات المكتبية العادية يمكننا استخدام كعكات HTTP أما في المستعرضات التي لا تدعم الكعكات فيتم استخدام ما يسمى بمُعرِّفات URL المطعّمة وهي عبارة عن عناوين URL تمت إضافة مُعرِّف فريد إليها ليميز الجلسة.

حالة الجلسة

توفر ASP.NET كما ذكرنا نسخة محسنة عن غرض Session المستخدم مسبقاً مع ASP والذي يسمح بإجراء المهام التالية:

- التعرف على المستخدمين من خلال مُعرِّف خاص بالجلسة.
- تخزين معلومات خاصة بمستخدم الجلسة.
- إدارة دورة حياة الجلسة من خلال طرق معالجة للأحداث.
- تحرير بيانات الجلسة بعد مضي زمن محدد.

هناك ميزتان أساسيتان لهذا الحل

- **الأولى:** عندما يتولى مخدم الاستجابة لطلبات متتالية من نفس الزبون حيث يمكن اعتماد مخدم محدد لتخزين معلومات

الجلسات مما يسمح بالحفاظ على الحالة أياً كان المخدم الذي يستجيب لطلب الزبون.

- **الثانية:** إمكانية تخزين معلومات الجلسة على قاعدة بيانات SQL Server مما يسمح باستعادة معلومات الجلسة في حال انهيار النظام أو إعادة تشغيل مخدم IIS.

أهم الطرق والخصائص التي يقدمها غرض Session هي:

الخاصة أو الطريقة	الوصف
الطريقة Abandon	تقوم بإهمال غرض Session الحالي وتحرير معلوماته.
الطريقة Add	إضافة عنصر.
الطريقة Clear	تقوم بمسح معلومات الجلسة ولكن لا يقوم بإهمال غرض الجلسة.
الطريقة Remove	تقوم بإزالة غرض من معلومات الجلسة الحالية.
الطريقة RemoveAll	تقوم بإزالة جميع العناصر من غرض الجلسة الحالية.
الطريقة RemoveAt	يقوم بإزالة العنصر ذي الدليل المحدد من غرض الجلسة الحالية.
الخاصة Count	تعيد عدد العناصر التي يتضمنها غرض الجلسة الحالية.
الخاصة IsCookieless	تعيد قيمة منطقية تحدد فيما إذا كانت الجلسة تدعم الكعكات أم لا.
الخاصة IsNewSession	تعيد قيمة منطقية تحدد فيما إذا كان الطلب هوالأول ضمن الجلسة.
الخاصة IsReadOnly	تعيد قيمة منطقية تحدد فيما إذا كانت معلومات الجلسة مخصصة للقراءة فقط.
الخاصة IsSynchronized	تعيد قيمة منطقية تحدد فيما إذا كانت الجلسة آمنة من حيث إمكانية خلق سياق برمجي إضافي لها.
الخاصة Item	تقوم بتعيين أو إعادة قيم عناصر الجلسة. تحديد قيمة العنصر بالشكل: Session[“keyName”]=value أو Session[index]=value
الخاصة keys	تعيد جميع المفاتيح الخاصة بغرض الجلسة الحالية.

استخدام غرض الجلسة

يمكننا التعامل مع غرض الجلسة ضمن النص البرمجي في الخلفية والخاص بالملف Global.asax أو ضمن النص البرمجي في

الخلفية لصفحة الوب المحمولة.

مثال:

يبين المثال التالي النص البرمجي في الخلفية للملف Global.asax والذي يأخذ الاسم Global.asax.cs. سنقوم في المثال باستخدام طريقتين لإضافة البيانات إلى غرض الجلسة:

- الأولى باستخدام مولد دليل بمفتاح UserStartTime لإضافة سلسلة محرفية تمثل زمن بداية غرض الجلسة.
- الثانية باستخدام الطريقة Session.Add لتعريف دخل يستخدم المفتاح HelpAccess والذي يحمل القيمة الأولية false.

```
using System;
using System.Collections;
using System.Web;
using System.Web.SessionState;

namespace MSPress.MobWeb.SessEx
{
    public class Global : System.Web.HttpApplication
    {
        protected void Session_Start(Object sender, EventArgs e)
        {
            Session["UserStartTime"]=DateTime.Now.ToLongTimeString();
            Boolean HelpAccess=false;
            Session.Add("HelpAccess",HelpAccess);
        }
    }
}
```

يتألف ملف Global.asax الذي يشير إلى النص البرمجي السابق من موجه Application@ وحيد:

```
<%@ Application Codebehind="Global.asax.cs"
    Inherits="MSPress.MobWeb.SessEx.Global" %>
```

نلاحظ في النص البرمجي التالي لصفحة نموذج الوب المحمولة، وجود عنصري تحكم Form:

- يوجد في النموذج الأول From1 زر وعنصر تحكم Label1 بحيث يجري تحديد قيمة Text لعنصر التحكم Label1 ضمن النص البرمجي في الخلفية.
- يوجد النموذج الثاني Form2 عنصري تحكم Label أحدهما يحدد كون هذه الصفحة هي صفحة مساعدة، أما الثاني فهوفارغ ويجري إسناد قيمة له في النص البرمجي العامل في الخلفية.

```
project <%@ Register TagPrefix="mobile"
Namespace="System.Web.UI.MobileControls"
    Assembly="System.Web.Mobile" %>
<%@ Page language="c#" Codebehind="MobileWebForm1.aspx.cs"
    Inherits="MSPress.MobWeb.SessEx.MobileWebForm1" %>

<mobile:Form id="Form1" runat="server">
    <mobile:Label id="Label1" runat="server"/>
```

```
    <mobile:Command id="Command1" runat="server">Go To
Help</mobile:Command>
</mobile:Form>
<mobile:Form id="Form2" runat="server">
<mobile:Label id="Label2" runat="server">
       This is a help page.
    </mobile:Label>
    <mobile:Label id="Label3" runat="server"></mobile:Label>
</mobile:Form>
```

أما النص البرمجي في الخلفية فهو:

```
using System;
using System.Web.Mobile;
using System.Web.SessionState;

namespace MSPress.MobWeb.SessEx
{
    public class MobileWebForm1 :
System.Web.UI.MobileControls.MobilePage
    {
        protected System.Web.UI.MobileControls.Label Label1;
        protected System.Web.UI.MobileControls.Label Label3;
        protected System.Web.UI.MobileControls.Command Command1;
        protected System.Web.UI.MobileControls.Form Form1;
        protected System.Web.UI.MobileControls.Form Form2;

        override protected void OnInit(EventArgs e)
        {
            InitializeComponent();
            base.OnInit(e);
        }

        private void InitializeComponent()
        {
            this.Load += new System.EventHandler(this.Page_Load);
            this.Command1.Click += new
                System.EventHandler(this.Command1_OnClick);
        }

        private void Page_Load(object sender, System.EventArgs e)
        {
            Label1.Text = "Help accessed: ";
            Label1.Text += Session["HelpAccess"].ToString();
        }

        private void Command1_OnClick(object sender, System.EventArgs
e)
        {
            //Switch to the Help form, set the flag in Session object
```

```
            Session["HelpAccess"] = true;
            Label3.Text = "Help accessed: ";
            Label3.Text += Session["HelpAccess"].ToString();
            ActiveForm = Form2;
        }
    }
}
```

عند تشغيل هذا التطبيق:

- يتم إرجاع القيمة False إلى المتحول HelpAccess ضمن غرض الجلسة.
- عند إظهار النموذج Form1 يجري تشغيل الطريقة Page_load وإظهار قيمة HelpAccess.
- عند استعمال المستخدم للزر على النموذج، تُرسل المعلومات إلى المخدم إضافة إلى تنفيذ طريقة معالجة الحدث Command_Click والتي ستؤدي إلى إسناد True إلى المتحول HelpAccess.
- عند حصول أي زيارة إلى Form2 من جديد ستظهر القيمة الجديدة لعنصر الجلسة HelpAccess وهي True.

استخدام الكعكات

تقوم Asp.NET بالتعرف على الجلسات من خلال مُعرِّف ID ضمن كعكة HTTP والتي يجري تمريرها بين الزبون والمخدم مع كل طلب واستجابة. تتطلب هذه العملية استخدام مُعرِّف فريد ومحمي من الوصول غير المشروع. وتؤمن ASP.NET هذا الأمر باستخدام مفتاح تشفير عشوائي يجري تغييره في كل مرة يُعاد فيها تشغيل مخدم الوب.

يكون طول مُعرِّف جلسة 32 بتاً مضافاً إليها محموعة من البيانات التي يتم تشفيرها للحصول على سلسة محارف بــ 16 بتاً تضمن هذه العملية أن يكون المُعرِّف ID فريد وتساعد في منع المخترقين من تخمين الخوارزمية المستخدمة لإنشائه.

توفر الكعكات طريقة فريدة لمخدم الوب للتعرف على المستخدمين المتصلين سلكياً كمستخدمي مستعرضات HTML. ولكن فرص هذه الآلية ضعيفة بالنسبة للمستخدمين اللاسلكيين كون أغلب التجهيزات المحمولة لا تدعم الكعكات. على أي حال، عند وجود احتمال دخول زبائن لا تدعم مستعرضاتهم الكعكات إلى تطبيقاتنا، يتوجب علينا استخدام مُعرِّفات URL المطعّمة عوضاً عنها.

استخدام مُعرِّفات URL المُطعّمة:

يمكننا تطعيم مُعرِّفات URL لتمرير محدد ID بين الزبون والمخدم بدلاً من استخدام الكعكات.
نستعرض في الصيغة التالية مثال على مُعرِّف URL مُطعَّم:
http://microsoft.com/myapp/(dcdb0uvhclb2b145ukpyrr55)/index.aspx

عند تلقي مخدم الوب الطلب يقوم بانتزاع الطعم منه وعندها يتم استخدام هذا المُعرِّف كما تستخدم الكعكات.

لاتُعتبر المُعرِّفات المطعّمة حلاً أمثلياً، إذ تواجه العديد من أنواع المستعرضات مشاكل في التعامل مع هذه المُعرِّفات وبالأخص في حالة المُعرِّفات النسبية وغير المطلقة. لذلك يجب المحافظة على خيار الكعكات متاحاً.

مثال:

يمكننا تعطيل الكعكات بسهولة بهدف استخدام مُعرِّفات URL المطعّمة كبديل وذلك بإسناد قيمة للواصفة cookieless التابعة للعنصر sessionState ضمن ملف Web.config كما يوضح المثال التالي:

```
<!-- configuration details -->
<sessionState
    mode="inProc"
    stateConnectionString="tcpip=127.0.0.1:42424"
    sqlConnectionString="data source=127.0.0.1;user id=sa;password="
   cookieless="true"
    timeout="20"
/>
<!-- more configuration details -->
```

لمنع معالج الحالة من تفعيل الكعكات يمكن وضع النص البرمجي المخصص لهذه الحالة ضمن الكتلة البرمجية التي يجري تنفيذها عندما تأخذ IsCookieLess القيمة True كما يلي:

```
if (Session.IsCookieless)
{

}
```

استخدام المتحولات المخفية

يلزمنا في بعض الأحيان تمرير كمية صغيرة من المعلومات بين صفحات الوب دون استخدام غرض Session.

فعند الحاجة على سبيل المثال إلى جمع معلومات عن طريق نموذج متعدد الأجزاء عادة وباستخدام HTML نلجأ إلى استعمل التأشيرة <input> مع الواصفة Type التي نسند إليها القيمة Hidden لتمرير المعلومات من جزء لآخر.

ولكن تختلف الأمور في WML إذ يجب التفكير مثلاً في وضع المتحولات ضمن خبىء المستعرض ثم إرسالها إلى المخدم حين انتهاء المستخدم من ملء جميع أجزاء النموذج.

تساعد الخاصة HiddenVariables للصف MobilePage في هذه الوظيفة حيث تسمح بتخزين ثنائيات مكونة من اسم وقيمة، ومن ثم يجري تمريرها بين المخدم والزبون كحقول مخفية.

نستطيع الوصول إلى الوظيفة المذكورة باستخدام غرض Session أيضاً لذا يمكن أن نتساءل عن دور المتحولات المخفية ومكان استخدامها. للإجابة على هذا التساؤل، يجب الانتباه إلى أن هذه الآلية تشكل أحد بدائل تمرير البيانات بين الصفحات ولكنها فعالة فقط عندما يكون حجم البيانات صغيراً بسبب استهلاكها لكم كبير من عرض الحزمة إضافة إلى محدودية الحجم الأقصى للمنصة التي

تدعمها التجهيزات المتوافقة مع WAP والذي لا يجب أن يتجاوز 1.4KB.

مثال:

يوضح النص البرمجي التالي استخدام تقنية المتحولات المخفية:

```
<%@ Register TagPrefix="mobile"
    Namespace="System.Web.UI.MobileControls"
    Assembly="System.Web.Mobile" %>
<%@ Page language="c#" Codebehind="MobileWebForm1.aspx.cs"
    Inherits="MSPress.MobWeb.HidVarEx.MobileWebForm1"
    AutoEventWireup="false" %>

<mobile:Form id="Form1" runat="server">
    <mobile:Label id="Label2" runat="server">Your name:</mobile:Label>
    <mobile:TextBox id="TextBoxName" runat="server"></mobile:TextBox>
    <mobile:Command id="Command1" runat="server">
        Submit
    </mobile:Command>
</mobile:Form>

<mobile:Form id="Form2" runat="server">
    <mobile:Label id="Label1" runat="server">
        Your e-mail:
    </mobile:Label>
    <mobile:TextBox id="TextBoxEmail" runat="server"/>
    <mobile:Command id="Command2" runat="server">
        Submit
    </mobile:Command>
</mobile:Form>

<mobile:Form id="Form3" runat="server">
    <mobile:TextView id="TextView1" runat="server">
        TextView
    </mobile:TextView>
</mobile:Form>
```

أما النص البرمجي في الخلفية فهو:

```
using System;
using System.Collections;
using System.Web;
using System.Web.Mobile;
using System.Web.SessionState;

namespace MSPress.MobWeb.HidVarEx
{
    public class MobileWebForm1 :
System.Web.UI.MobileControls.MobilePage
    {
        protected System.Web.UI.MobileControls.Form Form2;
```

```
        protected System.Web.UI.MobileControls.Command Command1;
        protected System.Web.UI.MobileControls.Command Command2;
        protected System.Web.UI.MobileControls.Form Form3;
        protected System.Web.UI.MobileControls.TextView TextView1;
        protected System.Web.UI.MobileControls.TextBox TextBoxName;
        protected System.Web.UI.MobileControls.TextBox TextBoxEmail;
        protected System.Web.UI.MobileControls.Form Form1;

        public MobileWebForm1()
        {
            Page.Init += new System.EventHandler(Page_Init);
        }

        private void Page_Init(object sender, EventArgs e)
        {
            InitializeComponent();
        }

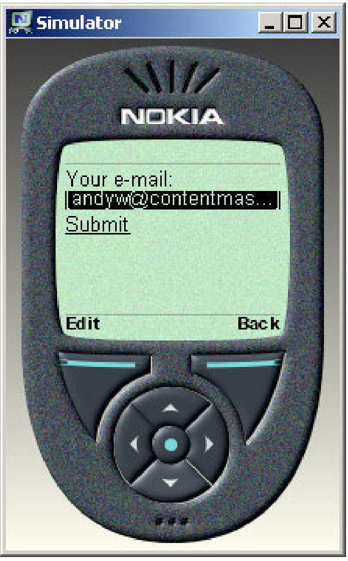
        private void InitializeComponent()
        {
            this.Command1.Click +=
                new System.EventHandler(this.Command1_Click);
            this.Command2.Click +=
                new System.EventHandler(this.Command2_Click);
            this.Form3.Activate +=
                new System.EventHandler(this.Form3_Activate);
        }

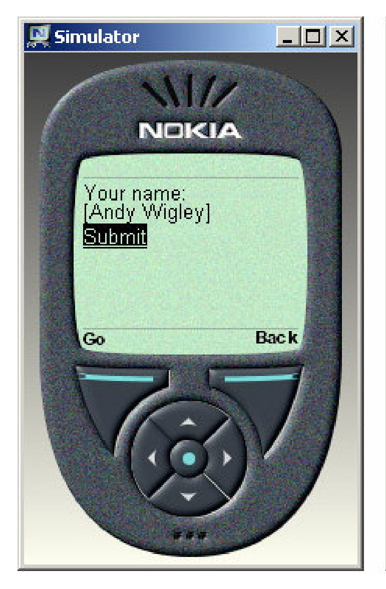
        private void Command1_Click(object sender, System.EventArgs e)
        {
            HiddenVariables.Add(TextBoxName.ID,TextBoxName.Text);
            this.ActiveForm=Form2;
        }
        private void Command2_Click(object sender, System.EventArgs e)
        {
            HiddenVariables.Add(TextBoxEmail.ID,TextBoxEmail.Text);
            this.ActiveForm=Form3;
        }

        private void Form3_Activate(object sender, System.EventArgs e)
        {
            String FormData="";
            foreach (Object o in HiddenVariables.Keys)
            {
                FormData+=o.ToString()+" "+HiddenVariables[o]+"<br>";
            }
            TextView1.Text=FormData;
        }
    }
}
```

وتكون النتيجة النهائية لتنفيذ هذا المثال على الشكل:







استخدام ViewState

توحي ASP.NET للمستخدم بإنطباع يفيد بإمكانية الحفاظ على الحالة للصفحات عبر عدة دورات بين المخدم والزبون. في الحقيقة لا تستمر معلومات الصفحة لأكثر من طلب أواستجابة، ولكن ما يحصل هوتخزين خصائص الصفحة والخاصة ViewState لكل عناصر تحكم المخدم فيها إلى مثل من الصف StateBag.

عندما يقوم المستخدم بإرسال طلب تجري عملية إعادة بناء الصفحة آلياً باستخدام قيم الخصائص ضمن مثل الصف StateBag.

يكون مجال رؤية الخاصة ViewState هوغرض MobilePage الحالي الذي يتضمن ملف aspx والبرنامج العامل في الخلفية. لذلك لن نستطيع استخدام الخاصة ViewState في حال كان التطبيق ينتقل من غرض MobilePage إلى آخر سواء باستخدام إعادة توجيه HTTP (باستخدام الطريقة (MobilePage.Response.Redirect("URL") أوبنقل التحكم إلى صفحة أخرى باستخدام الطريقة (MobilePage.Server.Transfer("URL").

يوضح المثال التالي كيفية حفظ خاصة ضمن viewState وكيفية استعادتها لاحقاً:

```
using System;
using System.Web;
using System.Web.Mobile;
using System.Web.UI.MobileControls;
using System.Web.UI;

public class MobileWebForm1 : System.Web.UI.MobileControls.MobilePage
```

```
{
    protected System.Web.UI.MobileControls.Command Command1;
    protected System.Web.UI.MobileControls.Label Label1;

    // MyMessage property get and set accessors
    // using the ViewState property
    public String MyMessage
    {
        get
        {
            // Explicit cast to String
            return (String) ViewState["MyMessage"];
        }
        set
        {
            ViewState["MyMessage"]=value;
        }
    }

    private void Command1_Click(object sender, System.EventArgs e)
    {
        // Consume the persisted property.
        Label1.Text=this.MyMessage;
    }
}
```

تكون الخاصة ViewState مفعّلة بشكل تلقائي في عناصر التحكم من جهة المخدم وصفحات الوب المحمولة. فإذا قمنا بإلغاء تفعيل الخاصة ViewState ضمن أحد عناصر التحكم سيتم تلقائياً إزالة تفعيلها بالنسبة لكل عناصر التحكم المحتواة ضمن هذا العنصر.

لإلغاء تفعيل عمل هذه الخاصة يكفي إسناد False إلى الواصفة EnableViewState ضمن تأشيرة عنصر التحكم من جهة المخدم كما يبين النص البرمجي التالي:

```
<mobile:Label id="Label1" runat="server" EnableViewState="False"/>
```

ولإلغاء تفعيل هذه الخاصة ضمن كامل الصفحة يكفي استخدام الصيغة:

```
<%@ Page language="c#" Codebehind="MobileWebForm1.aspx.cs"
    Inherits="MobileWebForm1" EnableViewState="False" %>
```

عادة ما يلجأ المطورون إلى إزالة تفعيل هذه الخاصة لتسريع عمل التطبيقات وذلك ضمن التطبيقات السلكية لأن هذه الطريقة (كما هي الحال مع استخدام المتحولات المخفية) تقوم بإرسال كم كبير من البيانات بين الزبون والمخدم مما يشكل عبء على الشبكة.

لما كانت هذه هي الحال بالنسبة إلى التطبيقات السلكية فالمنطقي أن يتحول استخدام هذه الآلية إلى مشكلة حقيقية في التطبيقات المحمولة عند أخذ محدودية عرض الحزمة بعين الاعتبار. لذلك تم تغيير آلية تطبيق ViewState المستخدمة ضمن عناصر التحكم المحمولة لتعتمد على تخزين البيانات ضمن غرض Session ولا تقوم في كل مرة بإرسال كامل المعلومات إلى الزبون بل ترسل فقط محدد تعريفي.

عند استخدام غرض Session لتخزين ViewState لا بد من أن نأخذ بعين الاعتبار نقطتين أساسيتين:

- الأولى تتعلق بزمن انتهاء حيث يجري عندها فقدان معلومات ViewState. يمكن تحديد زمن انتهاء صلاحية الجلسة بتعيين قيمة للواصفة timeout ضمن ملف التطبيق Web.config كما هوموضح في النص التالي :

```
<configuration>
    <system.web>
        <sessionState
            mode="inProc"
            cookieless="true"
        timeout="20"
        />
    </system.web>
</configuration>
```

- أما النقطة الثانية فتتعلق بالمعلومات المخزنة في الجلسة وتطابقها مع الصفحة الحالية، إذ قد نرى سيناريومن عدم المزامنة بين قيمة ViewState المخزنة في غرض الجلسة والصفحة، فإذا انتقل مستخدم من صفحة إلى أخرى ثم استخدم زر Back وعاد إلى الصفحة السابقة ستكون قيمة ViewState هي تلك الخاصة بالصفحة الثانية وليس الحالية.

تم حل هذه المشكلة باستخدام الواصفة sessionStateHistorySize الخاصة بعنصر mobileControls ضمن ملف Web.config حسب ما يظهر في النص البرمجي التالي:

```
<configuration>
    <system.web>
        <mobileControls sessionStateHistorySize="10"/>
    <system.web>
</configuration>
```

حالة التطبيق

يعرَّف التطبيق في ASP.NET بأنه كتلة جميع الملفات التي يقوم محرك زمن التشغيل باستخدامها أوتشغيلها ضمن مجال الرؤية الخاص بالمجلد الافتراضي وجميع المجلدات الفرعية ضمنه. ويحتاج المطور في بعض الأوقات إلى توليد أمثال عن المتحولات والأغراض التي تمتلك مجال رؤية يغطي التطبيق عوضاً أن يكون محدوداً على مستوى الجلسة. لذا يوفر الصف System.Web.HttpApplicationState الأمكانية المذكورة للمطورين.

يمثل الغرض Application التطبيق بحد ذاته ويتم إنشاؤه حال إرسال أول طلب إلى ملف ضمن المجلد الافتراضي الخاص بالتطبيق، ويمكن الوصول إليه باستخدام الخاصة Application للصف System.Web.HttpApplication والخاصة Application للصف MobilePage.

يمتلك غرض Application نفس الطرق التي رأيناها ضمن الغرض Session لذلك يمكن استخدامه بطريقة مشابهة مع مراعاة فكرة كون المعلومات التي يخزنها غرض Application تبقى مخزنة بين الطلبات المرسلة من عدة مستخدمين وتكون متوفرة لجميع مستخدمي التطبيق.

يبقى الغرض Application ضمن ذاكرة المخدم حتى يقوم المستخدم بإيقاف عمل المخدم أوبتعديل أوتحديث ملف Global.asax.

استخدام حالة التطبيق ضمن الملف Global.asax:

يمكنك التعامل مع البيانات ضمن غرض application باستخدام النص البرمجي في ملف Global.asax الذي يتوضع ضمن الملف الجذر في المجلد الافتراضي. يمكننا أيضاً العمل مع معلومات حالة التطبيق من خلال الوحدة النمطية العاملة في الخلفية للنص البرمجي Global.asax.

يوضح النص البرمجي التالي استخدام غرض Application ضمن الملف Global.asax:

```
using System;
using System.Collections;
using System.ComponentModel;
using System.Web;

namespace MSPress.MobWeb.AppObjEx
{
    public class Global : System.Web.HttpApplication
    {
        protected void Application_Start(Object sender, EventArgs e)
        {
            // Declare and assign a value to the global variable.
            String AppStartTime = DateTime.Now.ToLongTimeString();
            // Add the global variable to the Application object.
            Application["AppStartTime"] = AppStartTime;
        }
    }
}
```

نلاحظ هنا أننا استخدمنا Application_Start كمعالج لحدث بدء عمل التطبيق. كما قمنا في بداية النص السابق بتحديد الملف الممثل للوحدة النمطية العاملة في الخلفية وذلك باستخدام :

```
<%@ Application Codebehind="Global.asax.cs"
    Inherits="MSPress.MobWeb.AppStateEx.Global" %>
```

يمكننا عندئذ الوصول إلى معلومات غرض التطبيق من أي صفحة. يوضح النص التالي حالة صفحة وب محمولة تحتوي نموذجين. النموذج الأول يقوم بسؤال المستخدم إدخال اسمه ثم يقوم بإرساله إلى المخدم عند ضغط زر الارسال. أما النموذج الثاني فسيعرض اسم آخر مستخدم قام بإدخال اسمه.

```
<%@ Register TagPrefix="mobile"
    Namespace="System.Web.UI.MobileControls"
    Assembly="System.Web.Mobile" %>
<%@ Page language="c#" Codebehind="MobileWebForm1.aspx.cs"

Inherits="MSPress.MobWeb.SharedApplicationStateExample.MobileWebForm1"
%>
<mobile:Form id="Form1" runat="server">
    <mobile:TextBox id="TextBox1" runat="server"></mobile:TextBox>
    <mobile:Command id="Command1" runat="server">Enter</mobile:Command>
</mobile:Form>

<mobile:Form id="Form2" runat="server">
    <mobile:Label id="Label1" runat="server">Label</mobile:Label>
</mobile:Form>
```

أما الوحدة النمطية في الخلفية فهي :

```
using System;
using System.Collections;
using System.Web;
using System.Web.Mobile;
using System.Web.SessionState;

namespace MSPress.MobWeb.SharedApplicationStateExample
{
    public class MobileWebForm1 :
System.Web.UI.MobileControls.MobilePage
    {
        protected System.Web.UI.MobileControls.Command Command1;
        protected System.Web.UI.MobileControls.TextBox TextBox1;
        protected System.Web.UI.MobileControls.Form Form2;
        protected System.Web.UI.MobileControls.Label Label1;

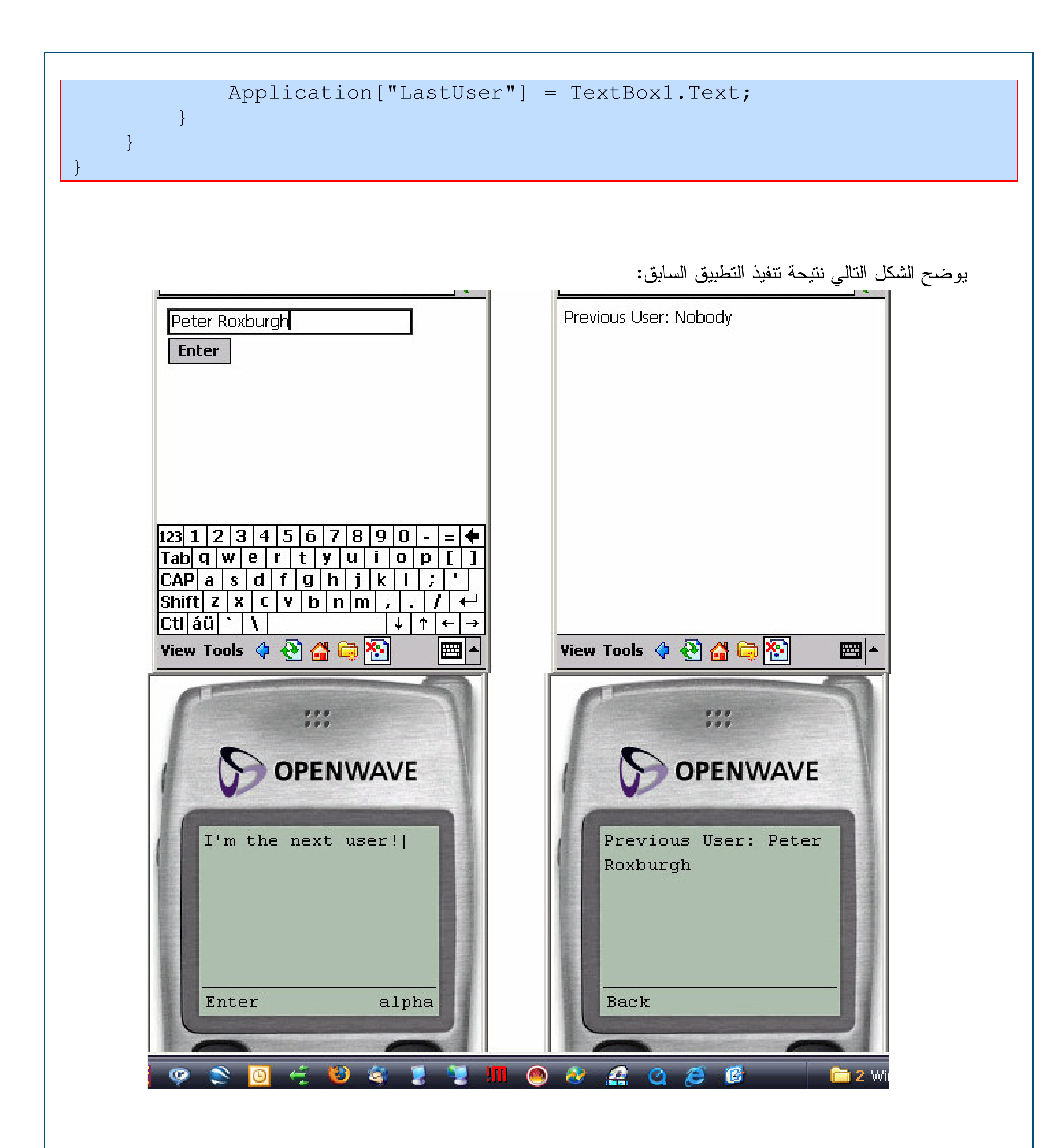

        override protected void OnInit(EventArgs e)
        {
            InitializeComponent();
            base.OnInit(e);
        }

        private void InitializeComponent()
        {
            Command1.Click += new
System.EventHandler(this.Command1_Click);
        }

        private void Command1_Click(object sender, System.EventArgs e)
        {
            ActiveForm = Form2;
            Label1.Text = "Previous User: " +
                Application["LastUser"].ToString();
```

```
                Application["LastUser"] = TextBox1.Text;
            }
        }
}
```

يوضح الشكل التالي نتيجة تنفيذ التطبيق السابق:



القسم الثاني عشر:

الموضوع الثاني: تحسين الأداء

الكلمات المفتاحية:

أداء، حالة، غرض.

ملخص:

إن أداء التطبيق وسرعته جزء أساسي من ضرورات الوصول إلى رضا المستخدمين عن هذا التطبيق، سنعالج في هذا الموضوع أهم النقاط التي قد تساعد على تحسين الأداء.

أهداف تعليمية:

يتعرف الطالب في هذا الفصل على:

- أهم النقاط الواجب مراعاتها لرفع أداء التطبيقات.
- أنواع الخبئ في ASP.NET.

تحسين أداء التطبيق

نُعرِّف التطبيق المناسب بأنه التطبيق الذي يحقق الوظائف بالشكل المطلوب ويحقق متعة العمل بحيث يكون بسيطاً، يتطلب أقل كم إدخال من المستخدم.

مهما تكن عملية تطوير التطبيق ذكية وتحقق الوظائف فإن التطبيق لن يحظى باهتمام المستخدمين إذا كان بطيئاً. لذلك سنحاول تغطية بعض الملاحظات التي تساعد على منح التطبيق أداء أعلى.

تحسين أداء التطبيق

- **إطفاء دعم إزالة العلل في النسخ التي تقوم ببنائها:**

الطريقة الأولى لتحسين الأداء هي التأكد من إطفاء دعم إزالة العلل في النسخ المبنية لأن وجودها سيؤدي إلى زيادة زمن الاستجابة كون محرك زمن التشغيل يقوم بمعالجة الطلبات باستخدام معالج آني. لإزالة تفعيل هذه الخاصة يجب إجراء التعديلات التي يظهرها النص البرمجي ضمن ملف Web.config

```
<configuration>
  <system.web>

    <compilation debug="false"/>

  </system.web>
</configuration>
```

- **إلغاء تفعيل ViewState في حال لم تكن ضرورية:**

ذكرنا أن استخدام ViewState يستهلك بعض الموارد أثنا عمل التطبيق لذلك فإلغاء تفعيله في حال عدم الحاجة إليه سيوفر هذه الموارد وبالتالي سيرفع الأداء.

- **إلغاء تفعيل حالة الجلسة عند عدم الحاجة إليها:**

إذا كان التطبيق لا يستخدم الوظائف التي يقدمها غرض الجلسة يُفضل إلغاء تفعيل عمل حالة الجلسة أيضاً.

```
<%@ Page EnableSessionState="false" %>
```

- **إجراء عملية خبئ للبيانات ضمن غرض Application:**

قد يشغل زبائن الوب المختلفون نفس التطبيق ويحاولون الوصول إلى نفس الموارد في نفس الوقت. أحد تقنيات التحسين المستخدمة في هذه الحالة هي تأمين مشاركة مجموعة المصادر المتكررة الاستخدام وغير المتغيرة بشكل كبير، للقراءة فقط وذلك بخبئ البيانات ضمن غرض Application عند تشغيل الطريقة Application_Start ضمن الملف Global.asax.

في هذه الحالة يقوم كل إجراء يُخدم زبون بالوصول إلى البيانات من خلال غرض Application دون الاضطرار إلى استعادة المعلومات من المصدر الأصلي. يوضح نص المثال التالي خبئ الغرض DataSet ضمن غرض Application وذلك من خلال الطريقةِ Appliaction_Start ضمن الملف Global.asax:

```
public void Application_Start()
{
    // Create the data component.
    AuthorsDataComponent.AuthsComponent myDataComp =
    new AuthorsDataComponent.AuthsComponent();

    // Use the data component to fetch a DataSet.
    AuthorsDataComponent.AuthsDataSet ds = myDataComp.AllAuthors;

    // Store the data source in the application state so that
    // the data source is available to all clients.
    Application["AuthsDataSet"] = ds;
}
```

عندها يمكن لأي غرض أن يصل إلى البيانات عن طريق غرض Application ضمن طريقة Page_Load للصفحة عوضاً عن جلب البيانات من قاعدة البيانات مرة أخرى.

```
void Page_Load(Object sender, EventArgs e)
{
    DataSet sourceDS = (DataSet)(Application["AuthsDataSet"]);

    List1.DataSource = sourceDS;
    List1.DataMember = "authors";

}
```

- **إطفاء دعم إزالة العلل في النسخ التي تقوم ببنائها:**

الطريقة الأولى لتحسين الأداء هي التأكد من إطفاء دعم إزالة العلل في نسخ المبنية لأن وجودها سيؤدي إلى زيادة زمن الاستجابة كون محرك زمن التشغيل يقوم بمعالجة الطلبات باستخدام معالج آني.

- **إلغاء تفعيل ViewState في حال لم تكن ضرورية:**

ذكرنا أن استخدام ViewState يستهلك بعض الموارد أثنا عمل التطبيق لذلك فإلغاء تفعيله في حال عدم الحاجة إليه سيوفر هذه الموارد وبالتالي سيرفع الأداء.

- **إلغاء تفعيل حالة الجلسة عند عدم الحاجة إليها:**

إذا كان التطبيق لا يستخدم الوظائف التي يقدمها غرض الجلسة يفضل إلغاء تفعيل عمل حالة الجلسة أيضاً.

- **إجراء عملية خبئ للبيانات ضمن غرض Application:**

قد يشغل زبائن الوب المختلفون نفس التطبيق ويحاولون الوصول إلى نفس الموارد في نفس الوقت. أحد تقنيات التحسين المستخدمة

في هذه الحالة هي تأمين مشاركة مجموعة المصادر متكررة الاستخدام غير المتغيرة بشكل كبير للقراءة فقط وذلك بخبئ البيانات ضمن غرض Application عند تشغيل الطريقة Application_Start ضمن الملف Global.asax.

في هذه الحالة يقوم كل إجراء يخدم زبون بالوصول إلى البيانات من خلال غرض Application دون الاضطرار إلى استعادة المعلومات من المصدر الأصلي.

تحسين أداء التطبيق
(تتمة)

€ استخدام التقسيم المخصص إلى صفحات:

في حالة تحميل أغراض DataSet الكبيرة ضمن عناصر تحكم SelectList, List وObjectList، أوعندما يتطلب توليد عناصر القائمة عمليات حساب معقدة، يجب التفكير باستخدام التقسيم المخصص إلى صفحات. في هذا النوع من التقسيم يجري تزويد العنصر حين إظهاره بالمحتويات فقط مما يوفر وقت الحساب الطويل لكامل محتوى عنصر التحكم.

€ عدم إجراء عمليات معالجة غير ضرورية عند عملية إعادة إرسال الصفحة :

من أهم تقنيات تحسين الأداء استخدام الخاصة MobilePage.IsPostBack لتجنب عمليات المعالجة غير الضرورية على البيانات، وذلك للتحقق من تنفيذ عمليات المعالجة في المرة الأولى لتحميل الصفحة وليس في كل مرة يتم إعادة إرسال الصفحة.

€ استخدام الطريقة System.Text.StringBuilder لجمع السلاسل المحرفية:

تعتبر هذه الطريقة أسرع من استخدام إشارة الجمع (+) لدمج السلاسل المحرفية وبالأخص في التطبيقات التي تتضمن الكثير من عمليات دمج سلاسل المحارف تلك.

```
StringBuilder detailText = new StringBuilder();
detailText.Append("This block of text ");
detailText.Append("will be <b>displayed</b> in a ");
detailText.Append("TextView Control.");

TextView1.Text = detailText.ToString();
```

€ ضبط الوصول إلى بيانات مخدم SQL بالشكل الأمثلي:

تساعد عملية استخدام الإجرائيات المخزنة للوصول إلى البيانات بشكل كبير في زيادة الأداء بشكل ملاحظ. كذلك هي الحال عند استخدام الأغراض SqlDataReader وOleDBDataReader في الحالات التي لا تتطلب وصول مخصص للكتابة أوالتحديث.

€ التصريح عن أنماط الأغراض في Visual Basic.NET

هناك طريقة أخيرة لزيادة أداء التطبيقات بالنسبة للمطورين الذين يستخدمون Visual Basic وهي إسناد true إلى الواصفة Strict التي تحدد ضرورة التصريح عن جميع أنماط الأغراض قبل استخدامها مما يوفر الوقت على محرك زمن التشغيل في التعرف على أنماط البيانات المختلفة المستخدمة.

```
<%@ Page Language="VB" Strict="true" %>
```

الخبئ

توفر ASP.NET ثلاث طرق للخبئ يمكن استخدامها لتحسين الأداء هي :

- خبئ الخرج.
- الخبئ المجزء.
- خبئ البيانات.

خبئ الخرج:

يمكننا استخدام الخبئ للصفحة باستخدام الموجه OutputCach@ في رأس صفحة نموذج الوب المحمول:

```
<%@ OutputCache Duration="60" VaryByParam="none"%>
```

تحدد الواصفة Duration عدد الثواني الذي يتوجب على IIS خبئ المعلومات فيه.

بوجود هذه العبارة في رأس الصفحة يحدد محرك زمن التشغيل فيما إذا كان يتوجب إرسال النسخة المخبئة أم إعادة توليد الصفحة بمقارنة سلسة محارف المحدد HTTP_User_Agent.

يجري إرسال سلسة المحارف هذه مع كل طلب لتعريف نمط المستعرض المستخدم. فإذا تم طلب صفحة ما مخبأة وهي مخصصة لنفس نوع المستعرض، ولم تنته صلاحيتها بعد، يجري إرسالها عوضاً عن إعادة توليدها من جديد.

يوضح المثال التالي هذه الفكرة:

```
<%@ OutputCache Duration="60" VaryByParam="none"%>
<%@ Register TagPrefix="mobile"
    Namespace="System.Web.UI.MobileControls"
    Assembly="System.Web.Mobile" %>
<%@ Page Inherits="System.Web.UI.MobileControls.MobilePage"
    Language="c#"%>

<html>
<head>
    <script language="c#" runat="server">
        public void Page_Load(Object sender, EventArgs e)
        {
            lblTime.Text = "Page Loaded at: " +
DateTime.Now.ToLocalTime();
        }
    </script>
</head>

<body
<mobile:Form runat="server" id="frmMain">
```

```
    <mobile:Label id="lblTime" runat="server"/>
</mobile:Form>
</body>
</html>
```

عند تشغيل هذا التطبيق وتحديث الصفحة لأكثر من مرة أومحاولة الوصول إلى الصفحة من نفس نوع المستعرض، لن يتغير الزمن الظاهر على هذه الصفحة لمدة 60 ثانية (أي لن يعاد توليد الصفحة في هذا الوقت). يجب أن يتضمن الموجه OutPutCache@ الواصفة VaryByParam ويمكن أن يتضمن بشكل كيفي الواصفات VaryByHeader و VaryByCustom وفيما يلي شرح لاستخدام كل منها:

- **VaryByParam**: استخدامها إلزامي كما رأينا. وهي تستخدم لتحديد قيمة أوأكثر ليتم إرسالها من الزبون إلى المخدم كمعامل عن طريق POST أو GET.
إذا تم تعيين قيمة هذه الواصفة إلى none سيقوم محرك زمن التشغيل بخبئ طلب GET الأولي إلا في حال تمرير المستخدم لمحددات أخرى عن طريق عنوان URL بعد الإشارة (?).
إذا أردت في هذه الحالة خبئ الصفحة الناتج باستخدام المعلومات المرسلة لا بد من تحديد اسماء المتحولات ضمن قيمة الواصفة VaryByParam مفصولة بفاصلة منقوطة كما يظهر في المثال:

```
<%@ OutputCache Duration="60" VaryByParam="selState;txtSearch" %>
```

ولجعل الخبئ يعتمد على جميع القيم المحددة ضمن سلسلة محارف الاستعلام أوضمن المعاملات المرسلة بواسطة POST يكفي إسناد القيمة (*) إلى هذه الواصفة.

```
<%@ OutputCache Duration="45" VaryByParam="*" %>
```

- **VaryByHeader**: تساعد هذه الواصفة في تفعيل الخبئ اعتماداً على عنصر ضمن رأس HTTP. فعلى سبيل المثال، لإجراء الخبئ اعتماداً على الرأس Accept-Language يجب كتابة موجه OutPutCache كما يلي:

```
<%@ OutputCache Duration="60"
VaryByHeader="Accept-Language"
VaryByParam="none" %>
```

- **VaryByCustom**: تستخدم هذه الواصفة لتغيير الخبئ اعتماداً على نمط المستعرض المستخدم أواعتماداً على سلسلة محرفية مخصصة يتم تحديدها.

تأخذ هذه الواصفة القيمة Browser بشكل تلقائي.

يمكننا تعيين مُعرِّف خبئ مخصص بإسناد القيمة المطلوبة إلى VaryByCustom وإعادة تعريف الطريقة GetVaryByCustomString للغرض HttpApplication ضمن ملف Global.asax.

يجب أن تعيد هذه الطريقة السلسلة المحرفية للطلب الحالي والتي سيتم استخدامها من قبل محرك زمن التشغيل لتغيير الخبئ وفقاً لها.

يوضح المثال التالي هذه الفكرة حيث يتم تحديد الموجه OutputCache@ كما يلي :

```
<%@ OutputCache Duration="60"
    VaryByCustom="MySelector"
VaryByParam="none" %>
```

كما يجب إعادة تعريف الطريقة GetVaryByCustomString ضمن الوحدة النمطية العاملة في الخلفية للملف Global.asax

```
public override string GetVaryByCustomString(
    HttpContext context, string arg)
{
    switch (arg){
        case "MySelector":
            // Send back the string that is used to distinguish
            // between client devices for output caching.
            return "MySelector=" + context.Request.Browser
                + context.Request.Frames;
        default:
            return "";
    }
}
```

الخبئ (تتمة)

استخدام الخبئ المجزء:

يُعرَّف الخبئ المجزء بأنه تقنية تُستخدم مع عناصر التحكم الخاصة بالمستخدم وهي مكونات ينشؤها المستخدم بقصد إعادة الاستخدام.

يمكن استخدام هذه الطريقة من الخبئ لتخزين الخرج الخاص بعناصر تحكم المستخدم والتي قد تشكل جزء من/أوكل صفحة الوب المرسلة إلى المستخدم.

من العناصر المرشحة لتستخدم هذا النوع من الخبئ: روابط الانتقال ضمن الموقع ورأس الصفحة والتذييل والأجزاء الأخرى التي يمكن بخبئها تحسين الأداء.

استخدام خبئ البيانات:

تدعم ASP.NET الخبئ عن طريق الذاكرة والذي يمكن أن يُستخدم لتخزين الأغراض المختلفة بين طلبات HTTP. يملك خبئ البيانات مجال رؤية يغطي التطبيق لذا يمكن الوصول إليه من أي جلسة ضمن التطبيق.

يمتلك خبئ البيانات بنية الفهرس لذا يكون التعامل معه برمجياً سهل جداً.

```
// Save DataSet in the data cache; has Application scope.
  Cache["DS"] = myDataset;

  // Get the DataSet from the cache.
  DataSet ds = (DataSet)(Cache["DS"])
```

القسم الثالث عشر:

دعم زبائن جدد

الكلمات المفتاحية:

زبون، موائم ، جهاز، ملف الإعداد، إمكانات

ملخص:

في معظم الحالات، تقوم ASP.NET بتشكيل صفحات نماذج الوب المحمول للزبائن بالشكل المتوقع، ولكـــن ASP.NET لاتـــدعم بعض المستعرضات الجديدة أوالنسخ الحديثة منها. سنستعرض في هذه الجلسة طرق دعم تجهيزات جديدة ولغات تأشير جديدة.

أهداف تعليمية:

يتعرف الطالب في هذا الفصل على:

- كيفية تعريف التجهيزات المحمولة ضمن ملفات الإعداد.
- التعرف على الجهاز وإمكاناته.
- ضبط وتعريف موائمات الأجهزة.

دعم زبائن جدد

في معظم الحالات تقوم ASP.NET بتشكيل صفحات نماذج الوب المحمول للزبائن بالشكل المتوقع، إلا أن بعض التجهيزات المحمولة تعطي أخطاء أثناء العمل على التطبيق، أو لا تظهر فيها صفحات التطبيق بالشكل المطلوب.

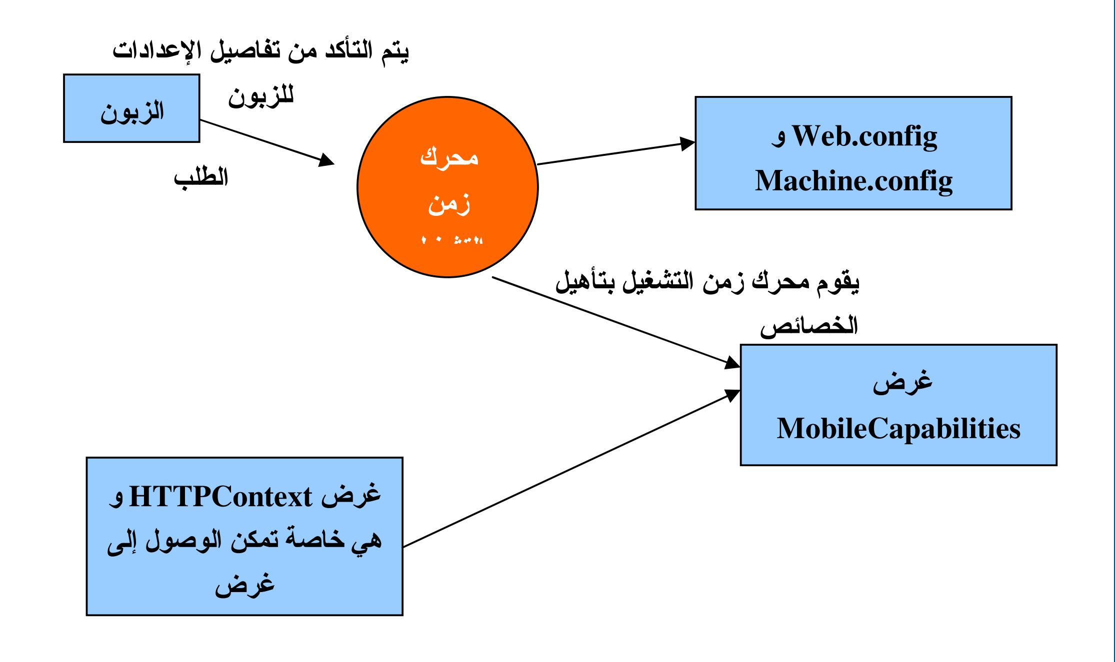
يعود سبب هذه الظاهرة إلى عدم دعم ASP.NET لمستعرض بسبب العدد الكبير من المستعرضات والعدد الكبير من النسخ المختلفة لكل مستعرض. إذ لن يكون من المفاجئ ألا تقدم ASP.NET دعماً لكل هذا التنوع الكبير من النسخ والمستعرضات.

على كل حال، تطلق Microsoft في كل فترة حزمة خاصة لدعم أنماط جديدة من المستعرضات وتحسين الدعم للمستعرضات المتوفرة. يمكن الوصول إلى هذه الحزمة عن طريق الارتباط
http://www.asp.net/mobile /deviceupdate.aspx?tabindex=6

لكن ماذا لو أردنا استخدام جهاز غير مدعوم بصورة رسمية بعد؟
لحسن الحظ تملك ASP.NET بنيان قابل للتوسع يسمح بإضافة أنواع جديدة من تجهيزات الزبون. لذا سنغطي خلال هذه الفقرة الطرق المستخدمة لتأدية هذا الغرض.

تعريف التجهيزات المحمولة ضمن ملفات الإعداد

لنتمكن من إضافة دعم لزبون جديد يجب أن نفهم كيف يتم التعرف على الزبائن وكيف يجري اختيار لغة التأشير الأنسب له.
يوضح الشكل التالي الإجراء الذي يحدد نمط الزبون ليقوم بدوره بتحديد تعريف له يكون متوفراً للتطبيق.



نلاحظ في الشكل السابق أن الزبون يقوم بإرسال طلب إلى صفحة نموذج الوب المحمول. عند تلقي الطلب يجري إنشاء غرض من الصف HttpRequest، ويقوم هذا الغرض بالوصول إلى الغرض MobileCapabilities عبر الخاصة Browser التي تُحفظ ضمنها معلومات حول قدرات الجهاز مرسل الطلب.

يجري بعدها فحص ملف الإعداد الخاص بالتطبيق Web.config وملف إعداد النظام Machine.config للقسم الخاص بالعنصر <browserCaps>.

إذا كنا قد ثبتنا التحديث بنسخته 2 أوأكثر فسنلاحظ أيضاً وجود هذا القسم ضمن الملف DeviceUpdate.config والذي يمكن إيجاده مع ملف machine.config ضمن المجلد /Microsoft.NET/Framework/version/CONFIG.

يوضح النص التالي القسم <browserCaps> ضمن ملفات الإعداد:

```
<browserCaps>
    <use var="HTTP_USER_AGENT" />
        <filter>
            <!-- Nokia -->
            <case
                match="Nokia.*">
                browser = "Nokia"
                mobileDeviceManufacturer = "Nokia"
                preferredRenderingType = "wml11"
                preferredRenderingMime = "text/vnd.wap.wml"
                preferredImageMime = "image/vnd.wap.wbmp"
                defaultScreenCharactersWidth = "20"
                defaultScreenCharactersHeight = "4"
                defaultScreenPixelsWidth="90"
                defaultScreenPixelsHeight="40"
                screenBitDepth = "1"
                isColor = "false"
                inputType = "telephoneKeypad"
                numberOfSoftkeys = "2"
                hasBackButton = "false"
                rendersWmlDoAcceptsInline = "false"
                rendersBreaksAfterWmlInput = "true"
                requiresUniqueFilePathSuffix = "true"
                maximumRenderedPageSize = "1397"
                canInitiateVoiceCall = "true"
                requiresPhoneNumbersAsPlainText = "true"
                rendersBreaksAfterWmlAnchor = "true"
                canRenderOneventAndPrevElementsTogether = "false"
                canRenderPostBackCards = "false"
                canSendMail = "false"

                <filter>
                    <case
                        match="Nokia7110/1.0 \((?'versionString'.*)\)">
                        type = "Nokia 7110"
                        version = ${versionString}
                        <filter
```

```
                              with="${versionString}"
                              match=
        "(?'browserMajorVersion'\w*)(?'browserMinorVersion'\.\w*).*">
                              majorVersion = ${browserMajorVersion}
                              minorVersion = ${browserMinorVersion}
                          </filter>
                          mobileDeviceModel = "7110"
                          optimumPageWeight = "800"
                          screenCharactersWidth="22"
                          screenCharactersHeight="4"
                          screenPixelsWidth="96"
                          screenPixelsHeight="44"
                      </case>
                  </filter>
              </case>
          </filter>
</browserCaps>
```

نلاحظ احتواء النص السابق على عدة أقسام <case> يحدد كل منها نوع من المستعرضات المدعومة. تستخدم هذه المقاطع التعابير النظامية لإجراء التطابق مع متحول البيئة HTTP_USER_AGENT الذي يجري تمريره عادة ضمن ترويسة طلب HTTP.

إذا تم إيجاد التطابق فيمكن عندها التعرف على المستعرض الزبون وإلا فسيتم تأهيل الغرض MobileCapabilities بالمعلومات التلقائية المحددة في الإعدادات التلقائية التي يمكن إيجادها في بداية القسم <browserCaps> ضمن ملف machine.config الذي يصنف الزبون على أنه مستعرض HTML 3.2 بنمط غير معروف.

تكون عملية البحث عن مطابق للزبون المطلوب بالترتيب ابتداء من machine.config إلى إلى DeviceUpdate.config إلى Web.config. بحيث تلغي معلومات الملف الذي ستتم قراءته أخيراً المعلومات السابقة.

يفضل عادة عدم وضع معلومات المستعرضات المختلفة ضمن machine.config نظراً لكونه ملف حساس، لذا يفضل إدراج أي تحديث أوتغييرات ضمن ملف DeviceUpdate.config بحيث يجري تحديد هذا الملف ضمن ملف machine.config وفق ما يلي:

```
<browserCaps>
    <result type="System.Web.Mobile.MobileCapabilities, System.Web.Mobile,
        Version=1.0.5000.0, Culture=neutral,
        PublicKeyToken=b03f5f7f11d50a3a" />
    <file src="deviceupdate.config" />
    <use var="HTTP_USER_AGENT" />
```

دعم الزبائن الجدد

تسهّل إعدادات ASP.NET المبنية كمقاطع وملفات XML عملية إضافة دعم لزبون جديد، حيث يمكن تعديل الملف machine.config أو deviceUpdate.config لتأمين دعم زبون جديد على كامل التطبيقات. أما إذا رغبنا إضافة هذا الدعم ليتناول تطبيق محدد فيجب استخدام الملف Web.config.

بغض النظر عن ملف الإعدادات الذي سنقوم باستخدامه هناك ثلاث خطوات يجب اتخاذها:

1- استخدام تعبير نظامي يسمح بالتعرف على الجهاز.
2- تعريف إمكانات الجهاز.
3- إدخال هذه الإمكانات ضمن ملف الإعداد.

التعرف على الجهاز

تُعتبر عملية اختبار ترويسة طلب HTTP User-Agent باستخدام تعبير نظامي، الطريقة الأفضل للتعرف على جهاز.

فعلى سبيل المثال تمتلك بعض التجهيزات مستعرض ثنائي الوضعية يمكنه معالجة HTML وWML بحيث يجري الفرز باستخدام ترويسة User-Agent الكافية للتمييز بينها. أما إذا كانت قيمة User-Agent واحدة لعدة مستعرضات، فيمكننا أولاً اختبار قيمة User-Agent ثم قيمة الترويسة Accept-Type لتعيين نوع المحتوى الذي يطلبه المستعرض.

يمكن الوصول إلى الترويسة User-Agent عن طريق الخاصة MobilePage.Request يقوم المثال التالي بكتابة قيمة الترويسة User-Agent إلى ملف header.log:

```
<%@ Import Namespace="System.IO" %>
<%@ Page language="c#"
Inherits="System.Web.UI.MobileControls.MobilePage"%>
<%@ Register TagPrefix="mobile"
    Namespace="System.Web.UI.MobileControls"
    Assembly="System.Web.Mobile" %>

<script runat="server" language="C#">
    public void Page_Load(object sender, System.EventArgs e)
    {
        FileStream fs = new FileStream(Request.PhysicalApplicationPath
+
                                       "header.log",
                                       FileMode.Append,
                                       FileAccess.Write);
        StreamWriter log = new StreamWriter(fs);

        //Write the user agent to the log file.
        log.WriteLine(Request.UserAgent);
        log.Flush();
        log.Close();
    }
</script>
<mobile:Form id="Form1" runat="server">
</mobile:Form>
```

بعد إيجاد قيمة الترويسة User-Agent لا بد لنا من إنشاء التعبير النظامي ضمن ملف الإعداد. فعلى سبيل المثال، إذا أردنا اختبار دعم جهاز معين للمستعرض EzWAP يمكننا أولاً إخطار النظام بضرورة استعمال قيمة HTTP_USER_AGENT في عملية الفلترة ثم تحديد الواصفة match للعنصر <case> كتعبير نظامي كما في الصيغة التالية :

```
<browsercaps>
    <use var="HTTP_USER_AGENT"/>
    <filter>
        <case match="EZOS - EzWAP 2.1 for Pocket PC">

        </case>
    </filter>
</browsercaps>
```

يمكننا توسيع التعبير النظامي ليشمل رقم النسخة أيضاً والذي يمكن استخدامه لتأهيل الخصائص في غرض MobileCapabilities:

```
<case match=
     "EZOS - EzWAP (?'majorVersion'\d+)(?'minorVersion'\.\d+)(\w*)"
  >
```

التعرف على إمكانات الجهاز

يملك غرض MobileCapabilities العديد من الخصائص التي تصف إمكانات الجهاز المحمول. يبين الجدول التالي أهم هذه الخصائص وأكثرها استعمالاً لهذا الغرض:

الخاصة	النمط	القيمة التلقائية	الوصف
Browser	String	Unkown	تعبر عن اسم المستعرض.
CanInitiateVoiceCall	Boolean	False	تأخذ هذه الخاصة القيمة True إذا كان الجهاز قادر على تأسيس اتصالات صوتية
CanSendMail	Boolean	False	تأخذ هذه الخاصة القيمة True إذا كان الجهاز قادر على إرسال البريد باستخدام الصيغة mailto
HasBackButton	Boolean	True	تأخذ هذه الخاصة القيمة True إذا كان الجهاز يوفر الزر Back مثل أكثر الهواتف المحمولة.
InputType	String	telephoneKeypad	تحدد نمط الكتابة على الجهاز اعتماداً على القيم الممكنة للخاصة telephoneKeypad أو VirtualKeyboard أو Keyboard

IsColor	Boolean	False	تأخذ هذه الخاصة القيمة True إذا كان الجهاز يدعم الألوان.
IsMobileDevice	Boolean	False	تأخذ هذه الخاصة القيمة True إذا كان الجهاز يدعم عناصر التحكم المحمولة.
MaximumRenderedPageSize	Int	2000	الطول الأقصى للصفحة التي يمكن إظهارها مقدراً بالبايت.
MaximumSoftLabelLength	Int	5	الطول الأقصى للنص الذي يمكن إظهاره ضمن الزر البرمجي.
MobileDeviceManufacturer	String	Unknown	اسم مُصنِّع الجهاز
MobileDeviceModel	String	Unknown	طراز الجهاز
NumberOfSoftKeys	Int	0	عدد الأزرار البرمجية التي يدعمها الجهاز
PreferredImageMime	String	Image/gif	نمط MIME المفضل الذي بإمكان الجهاز إظهاره.
PreferredRenderingType	String	Html32	\ النمط المفضل للمحتوى والذي يمكن للجهاز إظهاره. تتضمن القيم المحتملة html32 , wml11 ,wml12, chtml10
ScreenBitDepth	Int	1	تعبر هذه الخاصة عن البعد اللوني للإظهار مقدراً ب bit/pixel
ScreenCharactersHeight	Int	6	الرقم التقريبي لسطور النص الذي يمكن للجهاز إظهاره
ScreenCharactersWidth	Int	12	الرقم التقريبي للمحارف التي يمكن للجهاز إظهارها على سطر واحد.
ScreenPixelsHeight	Int	72	ارتفاع شاشة الإظهار بالبيكسل.
ScreenPixelsWidth	Int	96	عرض شاشة الإظهار بالبيكسل.

إذا تم تعيين الخصائص السابقة بشكل صحيح ضمن الإعدادات، فسيجري تحميل التطبيقات المحمولة بصورة مقبولة على الجهاز. ولكن كيف يمكننا معرفة هذه المعلومات؟

هناك عادة ثلاث طرق :

1- عن طريق الوثائق المرفقة بالجهاز.

2- عن طريق تطوير برنامج صغير بلغة تأشير يدعمها الجهاز ومحاولة استخلاص قيم هذه الخصائص.

3- عن طريق تغيير القيم ضمن ملف الإعداد والتأكد من عمل كل منها (الطريقة عادة غير فعالة نظراً لما تستهلك من وقت كبير)

استخدام لغة تأشير للتعرف على إمكانات الجهاز

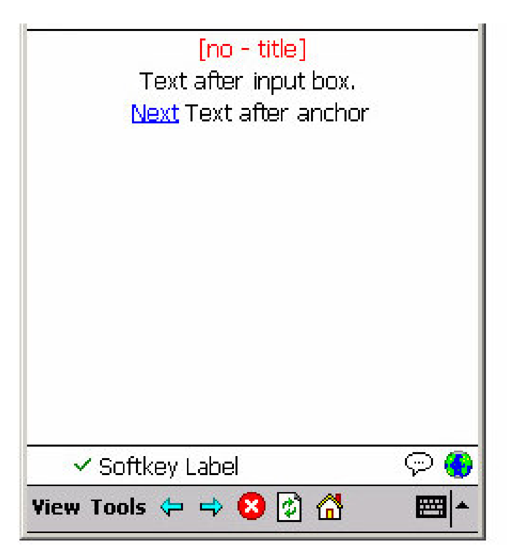
تتلخص الفكرة هنا في محاولة كتابة نص بلغة تأشير مع مراعاة دلالات قيم خصائص الغرض MobileCapabilities. فإذا أخذنا النص التالي المكتوب بلغة WML :

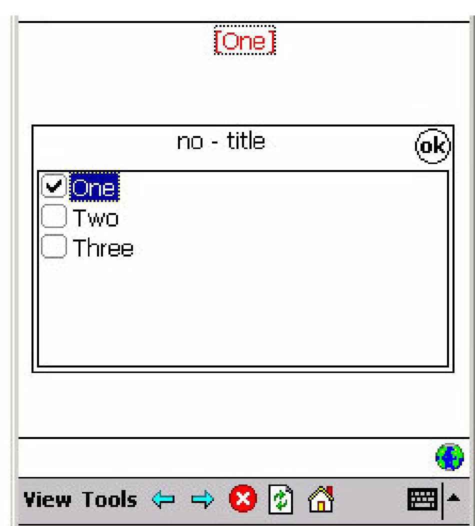
```
<% Response.ContentType= "text/vnd.wap.wml" %>
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.1//EN"
    "http://www.wapforum.org/DTD/wml_1.1.xml">

<wml>
    <card id="card1" title="Card #1" newcontext="true">
        <do type="accept" label="Softkey label">
            <go href="#card2"/>
        </do>
        <p align="center">
            <input type="text" name="test"/>
            Text after input box.
            <br/>
            <a href="#card2" title="anchor label">Next</a>
            Text after anchor
        </p>
    </card>

    <card id="card2" title="Card #2">
        <p align="center">
            <select>
                <option>One</option>
                <option>Two</option>
                <option>Three</option>
            </select>
        </p>
    </card>
</wml>
```

يظهر المستعرض EZWAP النتيجة التالية لهذا النص:





نلاحظ من النتيجة أن:

- المستعرض قام بإظهار سطر بعد حقل الإدخال لكنه لم يُظهر السطر الإضافي بعد الارتباط.
- تم إظهار علامة خاصة بالعنصر <do> في أسفل الصفحة كزر برمجي وتم استيعاب طول العبارة المخصصة ضمنه لذا.
- عند الدخول إلى البطاقة الثانية تم إظهار وصلة عند النقر عليها تم إظهار لائحة لاختيار.

من خلال النتائج التي رأيناها يمكننا أن نتوقع القيم التالية لخصائص MobileCapabilities

الخاصة	القيمة
MaximumSoftKeyLabelLength	20
RendersBreaksAfterWMLAnchor	false
RendersBreaksAfterWMLInput	false
RendersWMLDoAcceptsInline	false
RendersWMLSelectAsMenuCards	false

تحديد إمكانات الجهاز عبر الاختبار

تُعتبر عملية تغيير قيمة الخاصة ضمن ملف الإعداد ومراقبة النتيجة على المستعرض، أحد الطرق التي يمكن اتباعها أيضاً لتحديد إمكانات الجهاز. تعتمد هذه الطريقة البسيطة آلية التجربة والخطأ.

سنقوم في مثال بسيط بتحديد إمكانات الجهاز عن طريق الاختبار. ففي النص البرمجي التالي سنختبر قيم للخصائص:
RenderBreakAfterWMLAnchor
.RenderBreaksAfterWMLInput

لنتمكن من القيام بهذا لا بد أن نبدأ من ملف إعداد يوفر الحد الأدنى من التفاصيل للتعرف على الجهاز ثم نقوم بتجربة الخرج:

```
<browserCaps>
```

```
    <use var="HTTP_USER_AGENT" />
    <filter>
        <case
            match=
     "EZOS - EzWAP (?'majorVersion'\d+)(?'minorVersion'\.\d+)(\w*)"
        >

            <!—start with previously established properties -->
            browser="EzWAP"
            type="EzWAP"
            version= ${majorVersion}.${minorVersion }
            majorVersion= ${majorVersion}
            minorVersion =${minorVersion }
            isMobileDevice="true"
            mobileDeviceModel="Pocket PC"
            preferredRenderingType="wml12"
            preferredRenderingMIME="text/vnd.wap.wml"
            preferredImageMIME="image/vnd.wap.wbmp"
            inputType="virtualKeyboard"

            <!—Test with default values for these properties first -->
            rendersBreaksAfterWMLAnchor="false"
            rendersBreaksAfterWMLInput="false"
        </case>
    </filter>
</browserCaps>
```

بالطبع لا بد لنا من إنشاء ملف صفحة وب محمول لنقوم باختبار الإعدادات بواسطته.

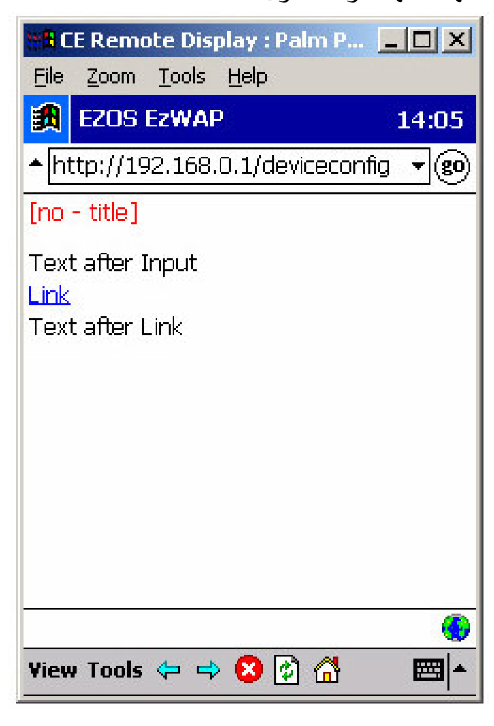
```
<%@ Register TagPrefix="mobile"
    Namespace="System.Web.UI.MobileControls"
    Assembly="System.Web.Mobile" %>
<%@ Page language="c#" Codebehind="MobileWebForm1.apsx.cs"
    Inherits="MSPress.MobWeb.TestBrowserCapabilities.MobileWebForm1"
    AutoEventWireup="false" %>

<mobile:Form id="Form1" runat="server">
    <mobile:TextBox id="TextBox1" runat="server"/>
    <mobile:Label id="Label1" runat="server">
        Text After Input
    </mobile:Label>
    <mobile:Link id="Link1" runat="server">
        Link
    </mobile:Link>
    <mobile:Label id="Label2" runat="server">
        Text After Link
    </mobile:Label>
</mobile:Form>
```

بمراقبة الخرج الخاص بنموذج الوب المحمول الذي قمنا بإنشاءه، نلاحظ وجود سطر إضافي بعد حقل إدخال النص وعندها يمكننا أن

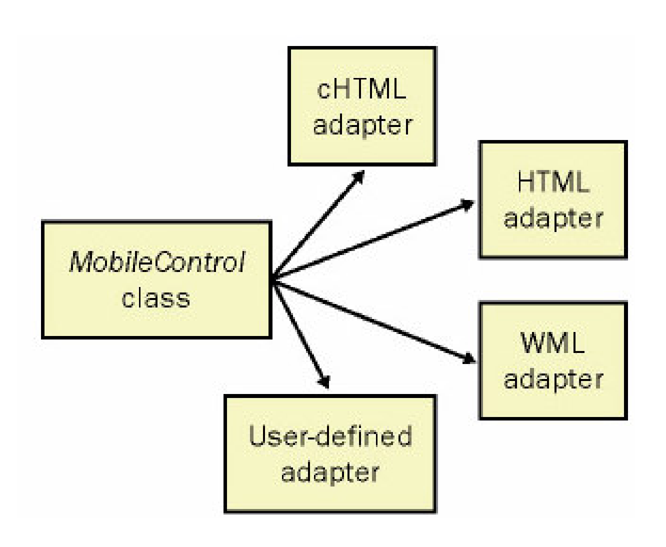
نعرف أنه يجب تعيين الخاصة RendersBreaksAfterWMLInput إلى false.

بمثل هذه الطريقة يمكن تعيين الكثير من قيم الخصائص ولكن كما نلاحظ فإن هذه الطريقة لا تتطلب معرفة بلغات التأشير المختلفة لكنها تستهلك وقتاً أطول.



ضبط موائمات الأجهزة

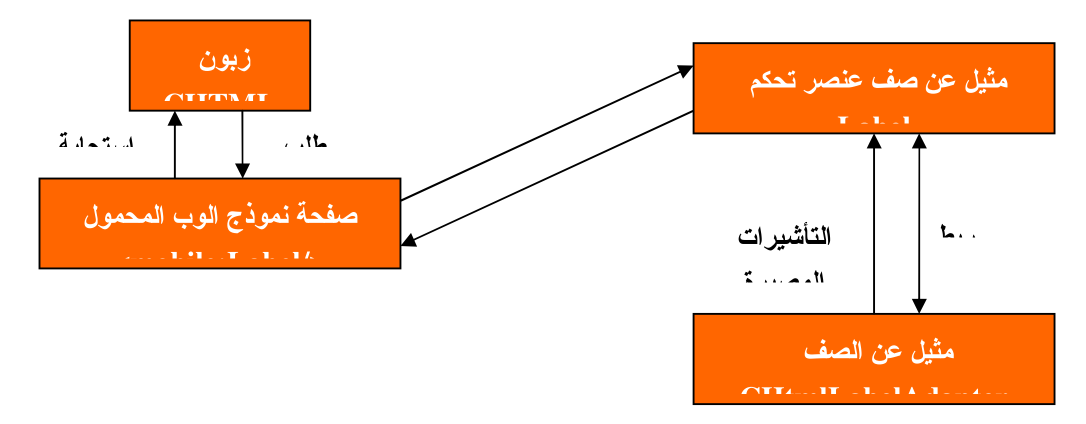
سنستعرض في هذا الجزء من الجلسة استعراض دور موائمات التجهيزات واكتشاف كيف يجري اختيار المجموعة الخاصة بموائم الجهاز عند ورود طلب ما وسوف نتعلم كيف نقوم بتوسيع المجموعة الخاصة بموائم الجهاز لدعم جهاز جديد أولغة تأشير جديدة.



بعد أن يقوم المحرك زمن التشغيل بالتعرف بنجاح على الجهاز ويجري تأهيل خصائص الغرض MobileCapabilities لا بد من تحديد لغة التأشير الأنسب للجهاز مرسل الطلب.

تعتبر موائمات الأجهزة الجزء من عناصر التحكم المحمولة الذي يقوم بتقديم عناصر التحكم المحمولة بالشكل المناسب على جهاز معين.

لذا يجب النظر إلى صفوف موائمات الأجهزة (مجموعاتها) على أنها جسر بين عنصر التحكم المجرد والزبون كما يظهر الشكل التالي.



تدعم عناصر التحكم المحمولة في ASP.NET مجموعة واسعة من الزبائن تشمل WML1.1 , WML1.2,cHTML1.0,HTML.32, XHTML. تقدم الموائمات المتوفرة نتائج جيدة على معظم التجهيزات المتوفرة. تم توفير النص المصدري لموائمات الأجهزة لتشجيع المطورين على خلق موائمات جديدة للأجهزة غير المدعومة. يمكن الوصول إلى النص المصدري عن طريق الوصلة *http://go.microsoft.com/fwlink/?LinkId=6350*.

استخدام مجموعات ضبط الموائمات الخاصة بالأجهزة

يمكننا تعريف العلاقة بين موائمات الأجهزة وعناصر تحكم نماذج الوب المحمول ضمن Web.config أو machine.config وذلك ضمن مجموعات.

فعلى سبيل المثال تم تزويد عناصر التحكم المحمولة في Asp.Net بثلاث أنماط من موائمات التجهيزات المسبقة التعريف كعناصر من الملف machine.config كعناصر <HtmlDeviceAdapters>، <WmlDeviceAdapters>، <ChtmlDeviceAdapter> كما تمت إضافة دعم <XHTMLDeviceAdapters>.

تقوم كل مجموعة موائمات أجهزة بمقابلة كل عنصر تحكم محمول بموائم الجهاز المناسب والذي يقوم بتصيير التأشيرات المناسبة إلى الزبون. عند تلقي طلب من الزبون، يجري إسناد موائم أوأكثر إلى هذا الطلب والذي سيقوم بدور تخديم هذا الطلب.

يجب تعيين مجموعة موائمات الأجهزة ضمن العنصر <mobileControls> ضمن Web.config أو machine.config حيث يدعم هذا العنصر عدة عناصر مشتقة من العنصر <device> المستخدم لتعريف مجموعات موائمات الأجهزة.

يظهر الجدول التالي خمس واصفات يدعمها العنصر <device>:

الواصفة	الوصف
name	اسم مجموعة الموائمات
predicateClass	اسم الصف الذي يحتوي طريقة الإسناد
predicateMethod	اسم طريقة الاسناد لمجموعة الموائمات. وهي عبارة عن طريقة ساكنة يجري استخدامها للتأكد فيما إذا كانت مجموعة موائمات الأجهزة مناسبة للجهاز الزبون الحالي.
pageAdapter	اسم صف موائم الصفحة الذي يقابل مجموعة الموائمات.
inheritsFrom	يمكن استخدامها لوراثة الإعدادات من مجموعة موائمات أجهزة أخرى.

يدعم العنصر <device> مجموعة عناصر أبناء من النوع <control> والتي تستخدم في إسناد عنصر تحكم محمول إلى موائم عنصر تحكم. فعلى سبيل المثال يمكننا إسناد عنصر التحكم المحمول Panel إلى موائم عنصر التحكم WMLPanelAdapter.

يملك عنصر <control> واصفتين هما name وadapter. يتم إسناد اسم صف عنصر التحكم المحمول إلى الواصفة name وإسم صف موائم عنصر التحكم إلى الواصفة adapter.

```
<?xml version="1.0" encoding="utf-8"?>
<configuration>
    <system.web>

        <!-- Other Web.config settings -->

        <mobileControls>
            <device
                name="HtmlDeviceAdapters"
                predicateClass=
                  "System.Web.UI.MobileControls.Adapters.HtmlPageAdapter"
                predicateMethod="DeviceQualifies"
                pageAdapter=
                  "System.Web.UI.MobileControls.Adapters.HtmlPageAdapter">

                <control name="System.Web.UI.MobileControls.Panel" adapter=
                    "System.Web.UI.MobileControls.Adapters.HtmlPanelAdapter"/>

                <control name="System.Web.UI.MobileControls.Form" adapter=
                    "System.Web.UI.MobileControls.Adapters.HtmlFormAdapter"/>

                <!—Adapter mappings continue ->

            </device>
        </mobileControls>
    </system.web>
</configuration>
```

تعريف مجموعة موائمات أجهزة

ذكرنا أنه يمكن لنا كتابة صفوف موائمات جديدة لعدة أسباب. فمثلاً يمكننا استبدال موائم جهاز قياسي بنسخة مخصصة أويمكن إضافة موائمات أجهزة لدعم عناصر تحكم مخصصة.

إذا كنّا نحاول استبدال موائم الجهاز القياسي بآخر مخصص، أوإضافة إعدادات لعنصر تحكم محمول مخصص ولموائماته، يمكننا تحديث تعريفات مجموعة موائمات التجهيزات ضمن ملف machine.config وذلك باستبدال اسم الصف الخاص بموائم الجهاز بالصف المقابل للنسخة الجديدة.

كحل بديل يمكن أيضاً تعريف مجموعة موائمات أجهزة ترث جميع الإعدادات من مجموعة موائمات باستخدام الواصفة inheritsFrom لعنصر <device>. إذ يتم تعريف عناصر أبناء <control> ضمن العنصر Device لتستبدل تلك التي تملك نفس الاسم في مجموعة الموائمات الآباء. كما يمكن تعريف عناصر <control> خاصة بعناصر التحكم المخصصة.

يبين المثال التالي الإعدادات الخاصة بوراثة مجموعة موائمات أجهزة WmlDeviceAdapters وتحديد طريقة اسناد باسم DeviceQualifies:

```
<?xml version="1.0" encoding="utf-8"?>
<configuration>
    <system.web>

        <!-- Other Web.config settings-->

        <mobileControls>
            <device
                name="NewWmlDeviceAdapters"
                inheritsFrom="WmlDeviceAdapters"
                predicateClass=

"System.Web.UI.MobileControls.Adapters.WmlPageAdapter"
                predicateMethod="DeviceQualifies"
                pageAdapter=

"System.Web.UI.MobileControls.Adapters.WmlPageAdapter">

                <control
                    name="System.Web.UI.MobileControls.MyControl"
                    adapter=

"System.Web.UI.MobileControls.Adapters.WmlMyControlAdapter"/>

            <!-- Place any new mappings here -->

            </device>
        </mobileControls>
```

```
    </system.web>
</configuration>
```

أما النص البرمجي الخاص بطريقة الاسناد فيكون:

```
public class Wml13PageAdapter : WmlPageAdapter
{
    public static bool DeviceQualifies(HttpContext context)
    {
        MobileCapabilities capabilities =
            ((MobileCapabilities)context.Request.Browser);
        bool qualifies = capabilities.Browser == "Openwave13";
        return qualifies;
    }
}
```

القسم الرابع عشر:

برمجة موائمات الأجهزة وإنشاء العناصر المخصصة من الصفر

الكلمات المفتاحية:

عنصر تحكم، موائم، جهاز، ملف الإعداد، عنصر مخصص.

ملخص:

سنتعرف في هذه الجلسة على دورة حياة عنصر التحكم المحمول وكيفية برمجة صفوف موائمات الأجهزة لتطبيق المنطق الخاص بكل جهاز.

أهداف تعليمية:

يتعرف الطالب في هذا الفصل على:

- دورة حياة عنصر التحكم والمراحل التي يمر بها
- دور موائمات الأجهزة
- كتابة النصوص المصدرية لموائمات الأجهزة
- بناء عنصر تحكم من الصفر وبناء موائمات الأجهزة المخصصة له
- ضبط وتعريف موائمات الأجهزة
- دعم التقسيم إلى صفحات والربط بقواعد البيانات لعناصر التحكم المخصصة.

برمجة موائمات الأجهزة وبناء عناصر التحكم

سنتعرف في هذه الجلسة على دورة حياة عنصر التحكم المحمول وكيفية برمجة صفوف موائمات الأجهزة لتطبيق المنطق الخاص بكل جهاز.

سنستعرض كذلك كيفية بناء عناصر التحكم بوراثة الصف MobileControl وكيفية تطبيق صفوف الموائمات عليه ليتم إظهاره على واجهة المستخدم.

بناء عناصر التحكم من الصفر:

إذا لم نجد ضمن عناصر التحكم المحمولة المتوفرة ما يلبي حاجتنا، فبإمكاننا تطوير عنصر تحكم محمول من الصفر وذلك بإنشاء صف يرث الصف MobileControl أو أي صف قاعدي مثل PageControl أو Panel.

يتطلب تطوير عناصر التحكم من الصفر حجم عمل أكبر من عملية إنشاء عناصر التحكم بالتركيب أوبالوراثة.

- فعند تطوير عنصر تحكم اعتماداً على عنصر آخر موجود، نقوم فقط بتعديل تصرفات عنصر التحكم الحالي عبر إضافة خاصة أوطريقة جديدة أوإعادة تعريف خصائص أوطرق موجودة.
- كذلك هي الحال عند تركيب عنصر تحكم باستخدام عدة عنصر تحكم من العناصر المتوفرة إذ لاتكون عملية التطوير معقدة أبداً وذلك بسبب وراثة جميع وظائف عناصر التحكم المُستخدمة.

أما في حالة تطوير عنصر تحكم من الصفر فلايتوجب علينا فقط تطوير الصف الخاص بعنصر التحكم، وإنما تطوير صف أوصفوف موائمات الأجهزة لعنصر التحكم الجديد هذا.

قبل الشروع بكتابة عناصر التحكم لا بد لنا من فهم المراحل التي يمر بها عنصر التحكم والطرق التي يقوم باستدعاءها وأوقات استدعائها.

يمكن تلخيص هذه المراحل كما يلي:

1- تحميل محتوى الصفحة وبناء عناصر التحكم.
2- استعادة حالة عناصر التحكم إذا كان قد جرى حفظ أي قيم سابقة في نهاية الطلب السابق للتطبيق.
3- معالجة البيانات التي تم إرسالها من الزبون وتحديث حالة عنصر التحكم.
4- إطلاق أحداث المخدم وتنفيذ معالجات الأحداث.
5- حفظ حالة عنصر التحكم بانتظار الطلب التالي.
6- تشكيل جواب وإرساله إلى الزبون.
7- التخلص من الصفحة وعناصر التحكم واستعادة الذاكرة المخصصة لها.

ذكرنا أيضاً أن إطار عمل يقوم باستدعاء مجموعة من الطرق التي تقوم باستبدال أوتطبيق الوظائف المخصصة المراد الوصول إليها.

يوضح الجدول التالي دورة حياة عنصر التحكم والطرق والأحداث التي يجري استدعاؤها:

المرحلة	ماذا يحدث	الطرق أوالأحداث التي يتم تنفيذها	طرق موائمات الأجهزة
التأهيل	يتم تأهيل الإعدادات اللازمة للطلب الحالي	OnInit: تقوم الطريقة OnInit بإطلاق الحدث Init وتقوم باستدعاء الطريقة OnInit التابعة	OnInit: يتم اختيار صف موائم الجهاز المناسب عند خلق مثل من عنصر التحكم. إذ تقوم

		لصف موائم الجهاز. لتطبيق منطق العمل لعنصر خاص يجب إعادة تعريف تلك الطريقة.	الطريقة OnInit ضمن الصف MobileControl باستدعاء الطريقة OnInit ضمن صف موائم الجهاز.
تحميل محتوى ViewState	يتم تحميل محتوى ViewStateمن الغرض المخزن في نهاية الطلب السابق.	:LoadViewState يجب أن نقوم بإعادة تعريف LoadViewState في حال رغبنا بتخصيص عملية استعادة الحالة. إذ تقوم الطريقة LoadPrivateViewStateبتحميل الحالة الداخلية التي تم إرسالها إلى المستخدم.	:LoadAdapterState إذا كان موائم الجهاز يحفظ أي معلومات خاصة بالجهاز ضمن ViewState، يجري تحميل هذه المعلومات في هذه المرحلة. تقوم الطريقة LoadPrivateViewState الخاصة بالصف MobileControl باستدعاء هذه الطريقة (LoadAdapterState)
إرسال البيانات من الزبون	يتم تحليل البيانات المرسلة من الزبون	:LoadPostData يتم إرسال نتيجة تفاعل المستخدم مع عنصر التحكم إلى المخدم كجزء من الطلب الحالي. يتم تحليل البيانات المرسلة وتحديث الخصائص المناسبة لهذا العنصر. لا تشكل العناصر التي لا تتبنى الصف IPostBackDataHandler جزءاً من هذه المرحلة.	بشكل تلقائي لا تشترك أي طرق تابعة لصف موائم جهاز في هذه المرحلة.
التحميل	تجري هنا الأعمال التي يجب أن تظهر عند كل طلب، كفتح اتصال بقاعدة البيانات مثلاً.	:OnLoad تقوم هذه الطريقة بإطلاق الحدث Load واستدعاء الطريقة OnLoad ضمن صف موائم الجهاز. في نهاية هذه الطريقة يكون عنصر التحكم مبني بصورة كاملة، وتطبح استعادة الحالة نتيجة من نتائج مرحلة إعادة إرسال المعلومات، كما تجري عملية استعادة البيانات من قواعد البيانات.	:OnLoad تقوم هذه الطريقة الخاصة بالصف MobileControl باستدعاء الطريقة OnLoad الخاصة بصف موائم الجهاز.
إرسال تنبيهات التغيير ضمن البيانات المرسلة	يتم إطلاق أحداث التغيير عند الحاجة، أي في حال غيرت البيانات المرسلة من بيانات الحالة الخاصة بعنصر التحكم.	:RaisePostDataChangeEvent يجري إطلاق أحداث التغيير إذا تغيرت بيانات الحالة بين عملية الإرسال السابقة والحالية، كما هي الحال عندما يكون الحدث TextChanged الخاص بعنصر التحكم TextBox أحد هذه الأحداث. لا تشترك عناصر التحكم التي لا تدعم	بصورة تلقائية، لا تشترك طرق صفوف موائم جهاز في هذه المرحلة.

		IPostBackDataHandler في هذه المرحلة.	
معالجة أحداث إرسال البيانات	الأحداث التي تستجيب من طرف المستخدم والناتجة عن عملية التفاعل مع الزبون.	RaisePostBackEvent: ينطلق هذا الحدث بعد أي حدث تغيير. يجري إطلاق ما يسمى حدث من جهة الزبون باتجاه المخدم. فعلى سبيل المثال عند نقر مستخدم على عنصر تحكم Command محمول تتم عملية إرسال المعطيات ويجري إطلاق الحدث Click. لا تشترك العناصر التي لاتتبنى الصف IPostBackEventHandler في هذه المرحلة.	HandlePagePostBackEvent: إذا أمكن للأحداث المرسلة إلى عنصر التحكم أن تتنوع اعتماداً على الجهاز الهدف، فيجب على عنصر التحكم في هذه الحالة استدعاء HandelPagePostBackEvent لإعطاء موائم الجهاز الفرصة لمعالجة حدث إرسال البيانات.
مرحلة ما قبل التعديل	تتضمن جميع التحديثات التي تتم قبل عملية تعديل عنصر التحكم.	OnPreRender: تقوم الطريقة onPreRender بإطلاق الحدث PreRender واستدعاء الطريقة OnPreRender ضمن صف موائم الجهاز. كما تتم في هذه المرحلة أي تحديثات إضافية على معلومات الحالة لعنصر التحكم.	تقوم الطريقة OnPreRender في الصف MobileControl باستدعاء الطريقة OnPreRender ضمن صف موائم الجهاز.
تخزين معلومات الحالة	يتم في هذه المرحلة حفظ معلومات الخاصة ViewState ضمن غرض من نمط سلسلة نصية ليتم حفظه.	عادة ما يكون الوضع الأصلي للطريقة SaveViewState مناسباً في معظم الحالات. على أي حال يمكن إعادة تعريف الطريقة SaveViewState عند الرغبة في تخصيص إدارة الحالة. تستدعي الطريقة SaveViewState الطريقة SavePrivateViewState والتي تقوم بتخزين معلومات الحالة الداخلية التي تمت إعادة إرسالها إلى الزبون. بسبب محدودية عرض الحزمة في الحلول المحمولة يجري تخزين ViewState ضمن غرض جلسة على المخدم كما شرحنا في جلسات سابقة. على أي حال، يمكن للمطور إلغاء تفعيل	SaveAdapterState: يمكن لغرض موائم الجهاز حفظ أي معلومات ViewState خاصة بالجهاز باستخدام هذه الطريقة. حيث تقوم الطريقة SavePrivateViewState الخاصة بالصف MobileControlباستدعاء هذه الطريقة.

		.ViewState تكون PrivateViewState مختلفة حيث يتم إرسالها إلى الزبون ولايمكن إلغاء تفعيلها.	
التعديل	في هذه المرحلة سيتم توليد الخرج الذي سيتم إرساله إلى الزبون.	Render تقوم هذه الطريقة باستدعاء الطريقة Render الخاصة بصف موائم الجهاز.	:Render تستخدم هذه الطريقة الغرض MobileTextWriter لتقوم بإخراج التأشيرات المطلوبة لعنصر التحكم على الجهاز الزبون.
التفريغ	في هذه المرحلة يقوم عنصر التحكم بتحرير جميع الموارد	OnUnload تقوم الطريقة OnUnload بإطلاق الحدث UnLoad وتقوم باستدعاء الطريقة OnUnload الخاصة بصف موائم الجهاز. يجب أن تحرر هذه المرحلة الموارد المكلفة مثل الاتصالات التي جرى تأسيسها مع قاعدة البيانات.	:OnUnLoad تقوم الطريقة OnUnload في صف MobileControl باستدعاء الطريقة OnUnLoad في صف موائم الجهاز.
الترتيب النهائي	يتم في هذه المرحلة التنظيف النهائي قبل إزالة العنصر من الذاكرة	Dispose	

دور موائمات الأجهزة

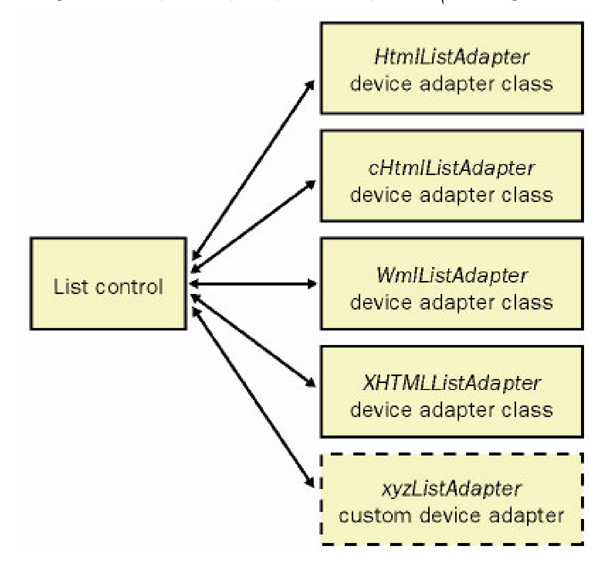
توسع تقنية ASP.NET المحمولة تقنيات ASP.NET الأخرى. وهي تعمل بنفس الطريقة وتستخدم مجموعة متشابهة من الطرق والتصرفات كون هذه العناصر كلها ترث من الصف System.Web.UI.Control. إلا أن إحدى الاختلافات الجوهرية بين عناصر التحكم المحمولة وتلك القياسية تكمن في استخدام عناصر التحكم المحمولة لموائمات الأجهزة.

إن أحد أول الطرق التي سنقوم بكتابتها عند تطوير عنصر تحكم مخصص في ASP.NET هي الطريقة Render والتي تقوم بتشكيل الخرج بصيغة لغة تأشير والذي سيتم إرساله إلى المستخدم.

يمتلك عنصر التحكم المحمول أيضاً الطريقة Render، إلا أنها لا تقوم بتشكيل خرج بل تقوم باستدعاء الطريقة Render المرتبطة بغرض موائم الجهاز والذي يقوم بدوره بتشكيل الخرج بلغة التأشير المناسبة.

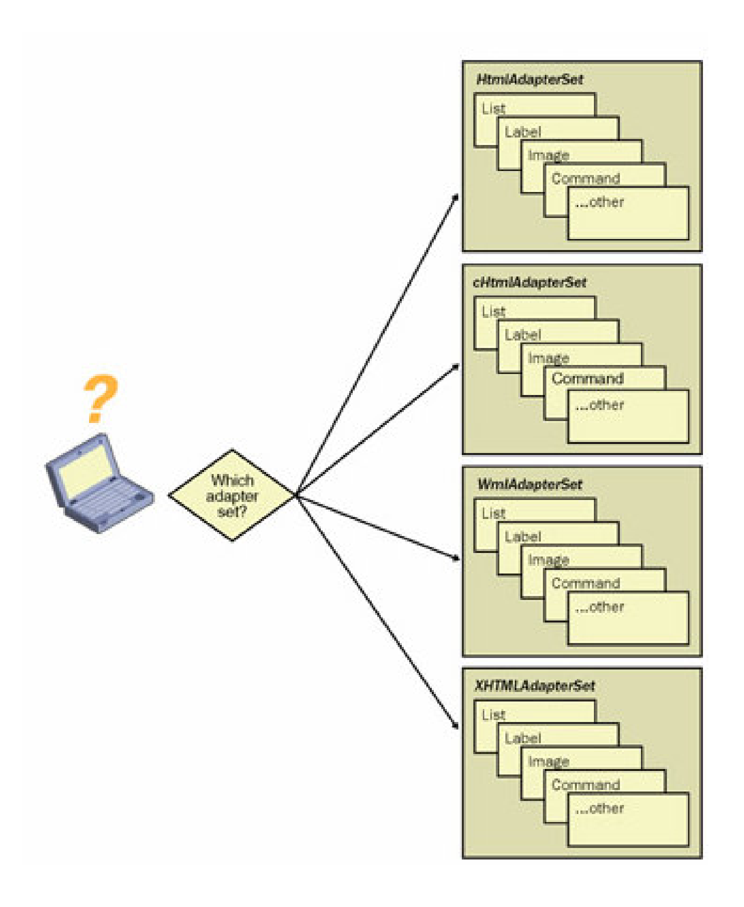
يحتاج كل عنصر تحكم محمول عادة إلى صف أوأكثر من صفوف موائمات الأجهزة حيث يوفر كل صف إمكانية العمل على نمط معين من الأجهزة.

يعبر الشكل التالي مثلاً عن التفاعل بين عنصر التحكم المحمول List وصفوف الموائمات المرتبطة به.



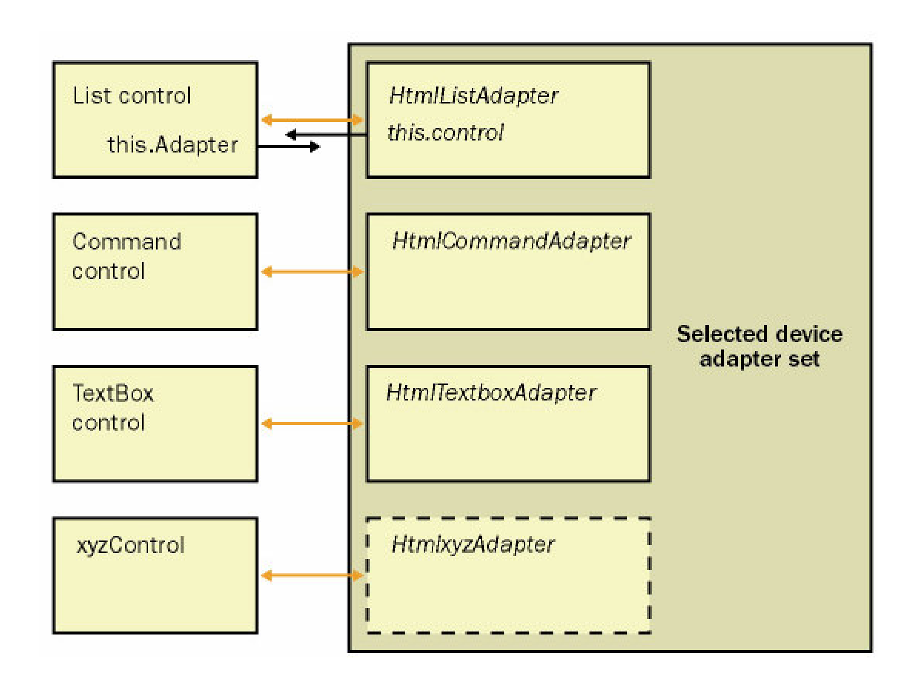
عندما يصل طلب مستخدم ما إلى المخدم، يجري التعرف على نوع الزبون بفحص السلسلة المحرفية User-Agent الممررة عن طريق ترويسة طلب HTTP كما رأينا في جلسة سابقة. ثم يتم البحث عن الجهاز ضمن لائحة التجهيزات المدعومة، ومن ثم يتم تخصيص مجموعة موائمات الأجهزة المناسبة لهذا الطلب.

تحتوي مجموعة موائمات الأجهزة، صف موائم جهاز خاص بكل عنصر تحكم محمول. تشترك جميع صفوف موائمات الأجهزة في مجموعة واحدة من حيث دعمها لنفس النمط من جهاز الزبون مثل WML1.1، HTML3.2، CHTML أو XHTML.



عند تطوير عناصر التحكم المحمولة الجديدة يجب تطوير موائمات أجهزة لكل نمط من الأجهزة المراد دعمها وإضافتها ضمن مجموعة الموائمات الخاصة بها.

عند معالجة طلب زبون ما سيعمل كل عنصر تحكم محمول ضمن الصفحة مع موائم الجهاز الخاص به ضمن مجموعة الموائمات المختارة لتوليد التأشيرات التي سيجري إرسالها إلى الزبون.



بنيان دعم الأجهزة القابل للتوسع

إن بنيان دعم الأجهزة في ASP.NET، قابل للتوسع. لذلك فإن عملية تحديث عناصر تحكم ASP.NET المحمولة سيؤدي إلى دعم أنواع جديدة من التجهيزات. وحتى إذا كنّا لا نريد انتظار التحديث من Microsoft وكان الموائم التلقائي الذي يجري إسناده إلى الجهاز الجديد غير مناسب، فبإمكاننا إضافة الدعم المطلوب بنفسنا.

العمل مع النصوص المصدرية لموائمات الأجهزة:

إنّ استخدام بيئة التطوير Visual studio.Net هي الطريقة الأكثر سهولة لتطوير موائمات الأجهزة. إذ يمكننا إضافة النص البرمجي لموائم الجهاز الذي قامت Microsoft بتقديمه إلى مشروع ما من الرابط http://go.microsoft.com/fwlink/?LinkId=6350 لتسهيل عملية التطوير. يمكن بعدها تحويل النص البرمجي إلى الصيغة المطلوبة ووضعه على مخدم الوب واستخدامه ضمن التطبيقات.

نورد في ما يلي المراحل التي يمكننا اتباعها لعملية التطوير:

1- إنشاء مشروع جديد بلغة #C (أو VB) من النوع Class Library تحت اسم المشروع Custom Adapters
2- حذف الصف التلقائي Class1.cs الذي قام Visual studio بإنشاءه ثم النقر بالزر الأيمن فوق المشروع في نافذة Solution Explorer واختيار Add ثم Add Existing Item ثم إضافة جميع نصوص #C البرمجية التي تم تحمليها من موقع Microsoft والخاصة بموائم الجهاز.
3- النقر بالزر الأيمن ضمن نافذة Solution Explorer واختيار Add Reference.
4- في لائحة NET. اختيار Mobile Internet Control Runtime. (System.Web.Mobile.dll) بالإضافة إلى مكونات System.Web.dll لضمها إلى المشروع.

5- يكون النص المصدري لموائمات الأجهزة هونفسه المستخدم لمعالجة موائمات الأجهزة المعرفة في فضاء الأسماء System.Web.UI.MobileControls.Adapter حيث تستخدم هذه النصوص نفس أسماء الصفوف. فإذا قمنا بعملية تجميع لملف يحتوي هذه النصوص وحاولنا استخدامه ستظهر رسالة خطأ لأن محرك زمن التشغيل لن يكون قادراً على التمييز بين الصف Sysytem.Web.UI.Mobile.Adapters.HtmlFromAdapter والصف ضمن ملفنا التجميعي والذي يملك نفس الاسم ويتوضع ضمن نفس فضاء الأسماء.

6- لحسن الحظ تستخدم النصوص المصدرية تستخدم عملية التجميع الشرطي لتعريف فضاء الأسماء الذي تنتمي إليه الصفوف. فإذا قام التطبيق بتعريف الثابت COMPILING_FOR_SHIPPED_SOURCE يجري تلقائياً تعريف مصادر موائم الجهاز ضمن System.Web.UI.MobileControls.ShippedAdapterSource.

7- في بيئة Visual Studio.NET يمكن تعريف هذا الثابت بالنقر بالزر الأيمن على المشروع ضمن نافذة Solution Explorer واختيار Properties، واختيار مجلد Configuration Properties ضمن شجرة المجلدات في أيسر صفحة الخصائص. ومن ثم اختيار الخيار Build، وأخيراً إدخال COMMPILING_FOR_SHIPPED_SOURCE بعد أي ثابت من الثوابت الموجودة ضمن المساحة المخصصة لثوابت التجميع الشرطي.

8- يجري إنشاء أي موائم جهاز جديد لعنصر التحكم المخصص وتعديل النص المصدري لعناصر التحكم الموجودة. ثم تجميع المشروع للحصول على ملف تجميعي جديد يحتوي كل موائمات الأجهزة.

9- لاستخدام الملف التجميعي الجديد يجب إنشاء مشروع وب محمول جديد، ثم النقر بالزر الأيمن على المشروع ضمن نافذة Solution Explorer واختيار Add Reference ثم اختيار المكتبة الحاوية على موائم الجهاز المخصص (DLL) وإضافتها إلى المشروع. سنحتاج أيضاً إلى ضبط إعدادات المشروع لاستخدام الموائمات المخصصة وذلك من ضمن الملف Web.config الخاص بالتطبيق.

بناء عنصر تحكم مخصص بسيط وبناء موائم جهاز

سنقوم في هذا الجزء من الجلسة ببناء عنصر تحكم شبيه بعنصر تحكم القائمة إضافة إلى إنشاء موائم جهاز لهذا العنصر.

بشكل تلقائي يتم إظهار عنصر التحكم هذا كجدول في الأجهزة التي تدعم HTML وكلائحة من الارتباطات في الأجهزة التي تدعم WML.

يكون عنصر التحكم الذي نبنيه عبارة عن عنصر تحكم جدول يتم إظهاره كجدول بعمودين على الأجهزة التي تدعم HTML وWML.

فيما يلي النص البرمجي لعنصر التحكم وهويساعد المستخدم في تحديد خاصة العنوان Title إضافة إلى خاصتين تمثلان العناصر ضمن العمودين في صف واحد.

```
using System;
using System.Web.UI.MobileControls;

namespace MSPress.MobWeb.CustomControls
{
    /// <summary>
```

```
    /// Simple example of a custom control built from scratch
    /// </summary>
    public class CMTable : MobileControl
    {
        private String _title, _item1Text, _item2Text;
        public CMTable()
        {
            Title = "";
            Item1Text = "";
            Item2Text = "";
        }

        /// <summary>
        /// Gets and sets the text that
        /// can be displayed as a title
        /// </summary>
        public String Title
        {
            get { return _title; }
            set {_title = value; }
        }

        /// <summary>
        /// Gets and sets the text displayed in column 1
        /// </summary>
        public String Item1Text
        {
            get { return _item1Text; }
            set {_item1Text = value; }
        }
        /// <summary>
        /// Gets and sets the text displayed in column 2
        /// </summary>
        public String Item2Text
        {
            get { return _item2Text; }
            set { _item2Text = value; }
        }
    }
}
```

يمكننا الآن استخدام Visual Studio لإنشاء مشروع من نمط مكتبة صفوف ثم إعطاء هذا المشروع الاسم CustomMobileLibrary مثلاً. لنقوم بعدها بإضافة مرجع إلى الملف التجميعي الخاص بزمن التشغيل لعناصر التحكم المحمولة (System.Web.Mobile.dll) كذلك System.Web.dll ثم إضافة ملف صف إلى المشروع والذي يحتوي النص المصدري السابق وأخيراً نقوم بتجميع الملف.

بعدها يجب إنشاء صفوف موائمات الأجهزة التي تحوي منطق الإظهار. ففي حالتنا يلزم صفان الأول مخصص لـــ HTML وCHTML والثاني مخصص لـــ WML

والنص البرمجي الخاص بموائم الجهاز لـــ HTML وCHTML:

```
using System;
using System.Web.UI.MobileControls;
using System.Web.UI.MobileControls.Adapters;
using MSPress.MobWeb.CustomControls;

namespace MSPress.MobWeb.CustomControls.Adapters
{
    public class HtmlCMTableAdapter : HtmlControlAdapter
    {
        protected new CMTable Control
        {
            get
            {
                return (CMTable)base.Control;
            }
        }

        public override void Render(HtmlMobileTextWriter writer)
        {
            String tableSuffix = "";
            Alignment alignment =
               (Alignment)Style[Style.AlignmentKey, true];
            if(alignment != Alignment.NotSet && alignment !=
Alignment.Left)
            {
                writer.Write("<div align=\"");
                writer.Write(alignment.ToString());
                writer.WriteLine("\">");
                tableSuffix = "\r\n</div>";
            }

            writer.AddAttribute("width","90%");
            writer.AddAttribute("cellpadding", "3");
            writer.RenderBeginTag("table");
            writer.WriteLine("");
            writer.Write("<tr><td>");
            writer.EnterFormat(Style);
            writer.WriteEncodedText(Control.Item1Text);
            writer.ExitFormat(Style);
            writer.WriteLine("</td>");
            writer.Write("<td>");
            writer.EnterFormat(Style);
            writer.WriteEncodedText(Control.Item2Text);
            writer.ExitFormat(Style);
            writer.WriteLine("</td></tr>");
            writer.RenderEndTag();
            writer.WriteLine(tableSuffix);
        }
    }
}
```

والنص البرمجي الخاص بموائم الجهاز لـــ WML:

```
using System;
using System.Web.UI.MobileControls;
using System.Web.UI.MobileControls.Adapters;
using MSPress.MobWeb.CustomControls;

namespace MSPress.MobWeb.CustomControls.Adapters
{
    public class WmlCMTableAdapter : WmlControlAdapter
    {
        protected new CMTable Control
        {
            get
            {
                return (CMTable)base.Control;
            }
        }
 public override void Render(WmlMobileTextWriter writer)
        {
            Alignment alignment =
                (Alignment)Style[Style.AlignmentKey, true];
            String alignID;
            switch (alignment)
            {
                case Alignment.Center:
                    alignID = "C";
                    break;
                case Alignment.Right:
                    alignID = "R";
                    break;
                default:
                    alignID = "L";
                    break;
            }

            //Write beginning of table
            writer.EnterLayout(Style);
            writer.EnterFormat(Style);
            writer.RenderText("<table", false, false);
            if (Control.Title.Length > 0)
                writer.WriteAttribute("title", Control.Title);
            writer.WriteAttribute("align", alignID + alignID);
            writer.WriteAttribute("columns", "2");
            writer.WriteLine(">");

            //First datacell
            writer.Write("<tr><td>");
            writer.RenderText(Control.Item1Text, true);
            writer.RenderText("</td><td>", false, false);
            //second datacell
            writer.RenderText(Control.Item2Text, true);
```

```
            writer.RenderText("</td></tr>", false, false);
            writer.WriteLine("</table>");

            //close table and output a trailing break
            writer.ExitFormat(Style);
            writer.ExitLayout(Style, true);
        }
    }
}
```

يجب تجميع هذه الصفوف ضمن ملف تجميعي بنفس الطريقة التي اتبعناها مع عنصر التحكم. يمكننا وضع هذه الصفوف ضمن ملفاتها التجميعية الخاصة أويمكن تضمينها في نفس المشروع المستخدم لعنصر التحكم ثم تحويل الكل إلى ملف تجميعي مرة واحدة.

كتابة لغة التأشير الخاصة بالجهاز باستخدام صفوف MobileTextWriter

يقوم إطار العمل الخاص بالصفحة باستدعاء الطريقة Render ممرراً كمعامل صف يتحدر من الصف System.Web.UI.MobileControls.Adapters.MobileTextWriter وهو WmlMobileTextWriter بالنسبة للموائم الذي يرث من WmlControlAdapter، و HtmlMobileTextWriter بالنسبة للموائم الذي يرث من الصف HtmlControlAdapter. تُستخدم هذه الصفوف لإرسال الخرج بلغة التأشير إلى الزبون.

تمتلك الصفوف المتحدرة من MobileTextWriter العديد من الطرائق التي تسهل إحراج لغة التأشير المطلوبة. فعلى سبيل المثال، تقوم WriteBeginTag("tagName") بفتح عنصر جديد وتقوم WriteAttribute("name","value") بإرسال خرج بواصفة ضمن عنصر ما. كما تقوم writeBreak بإرسال خرج تأشيرة
، في حين تقوم كلٌ من WriteText("text") و WriteEncodedText("text") بكتابة نص إلى الخرج بخيار إما باستخدام ترميز لتمثيل المحارف الخاصة أوبصورة مباشرة.

معالجة واصفات الإظهار ضمن موائمات الأجهزة:

تدعم جميع عناصر التحكم المحمولة نفس المجموعة من خصائص الإظهار الأسلوب مثل ForeColor، BackColor، Font...
وتكون معظم واصفات الإظهار استشارية بمعنى أنه سوف يتم احترامها وتطبيقها في حال ثبت دعم الجهاز لها.
يقرر النص البرمجي الخاص بصف موائم الجهاز إمكانية ترجمة خاصة الإظهار التي قام المطور باستخدامها وذلك بالتأثير ضمن الخرج المرسل إلى المستخدم.
من البديهي مثلاً عند إسناد القيمة Red إلى الخاصة ForeColor مثلاً أن تعمل بشكل طبيعي على مستعرض HTML ملون ولكن لن يكون لها أي تأثير على مستعرض WML1.1 وحيد اللون. تسهل صفوف MobileTextWriter إرسال خرج يميز العناصر المطلوبة باستخدام لغة التأشير المناسبة.
تكون الطرائق الموضحة أدناه للصف MobileTextWriter هي الأكثر استخداماً لتطبيق هذا التمايز في الخرج.

- EnterLayout(Style style): تقوم ببدء كتلة مقطع جديد ويُطبق عليها الأسلوب المتضمن في غرض style.
- ExitLayout(Style style) : تقوم بإغلاق كتلة مقطع.
- EnterStyle(Style style) : تقوم بكتابة تأشيرة الفتح.

€ ExitStyle(Style style): تقوم بكتابة تأشيرة الإغلاق.

ذكرنا أنه إذا قمنا بإنشاء موائم جهاز جديد لجهاز WML يجب الوراثة من الصف System.Web.UI.MobileControls.WmlControlAdaper أما إذا كنّا نريد إنشاء موائم HTML فيجب الوراثة من الصف System.Web.UI.MobileControlAdapter.

تكون الطرق EnterLAyout، ExitLayout، EnterStyle، ExitStyle ضمن موائم WmlControlAdapter و HtmlControlAdapter على علم بمحدودية الصف الخاص بالجهاز الذي تعمل عليه، لذلك فهي تعيد فقط التأشيرات المناسبة دون تضمين الواصفات التي لن تعمل على هذه التجهيزات. في معظم الحالات ولضمان ضبط واصفات الإظهار المناسبة عند إرسال الخرج إلى الجهاز يكفي فقط استدعاء الطريقة MobileTextWriter وتمرير خاصة Style إليها بشكل مشابه لما يلي :

```
writer.EnterLayout(this.Style);  // can leave off the 'this.' for
brevity
```

إظهار النص بعد استخدام EnterLayout و EnterStyle:

يجب علينا دائماً ضمن موائمات تجهيزات WML استخدام الطريقة RenderText لتخريج الـتأشيرات مباشرة بعد استدعاء EnterLayout و EnterStyle.

تقوم طريقة RenderText باستدعاء الطرق المسؤولة عن إرسال التأشيرات المنسقة إذا كانت مطلوبة.

فعلى سبيل المثال، يقوم النص البرمجي التالي بإخراج مقطع وتأشيرات تنسيق كما نتوقع:

```
writer.EnterLayout(Style);
writer.EnterFormat(Style);
writer.WriteBeginTag("table");
```

ولكن ما يجب استخدامه فعلياً هو:

```
writer.EnterLayout(Style);
writer.EnterFormat(Style);
writer.RenderText("table", false, false);
```

ملاحظة هامة : لا ينطبق هذا القيد على الغرض HtmlMobileTextWriter بل على الغرض WmlMobileTextWriter فقط.

يمكننا الاستعلام عن واصفة محددة للأسلوب باستخدام غرض Style لعنصر التحكم حيث يحدد هذا الغرض مجموعة من الثوابت مثل Style.AligmnetKey، Style.BoldKey و Style.ItalicKey. يمكن استخدام هذه الثوابت كدليل لخصائص الغرض Style كما في الصيغة:

```
Alignment alignment =  (Alignment)Style[Style.AlignmentKey, true];
```

تشير قيمة المعامل الثاني في حالة كانت قيمتها (False) إلى إمكانية استعادة القيمة المطبقة مباشرة على عنصر التحكم، أنا إذا كانت قيمتها True فتشير إلى إمكانية استعادة القيمة الموروثة من عنصر التحكم.

استخدام عناصر التحكم المخصصة وموائمات الأجهزة:

رأينا في بداية هذه الجلسة كيف يمكن إنشاء عنصر تحكم مخصص وموائمات أجهزة خاصة بهذا العنصر. سنقوم الآن باستعراض مثال عن كيفية استخدام عنصر التحكم هذا في صفحة نموذج وب محمول.

يوضح المثال التالي استخدام عنصر التحكم المخصص CMTable :

```
<%@ Register TagPrefix="CMcustom"
Namespace="MSPress.MobWeb.CustomControls"
    Assembly="CustomMobileControlLibrary" %>
<%@ Register TagPrefix="mobile"
    Namespace="System.Web.UI.MobileControls"
    Assembly="System.Web.Mobile" %>
<%@ Page language="c#"
    Inherits="System.Web.UI.MobileControls.MobilePage" %>

<mobile:form id="Form1" runat="server" Alignment="Center">
    <CMcustom:CMTable id="CmTable1" title="A title" runat="server"
        StyleReference="title" Font-Size="Small"
        Item1Text="Simple" Item2Text="Table" >
    </CMcustom: CMTable>
    <CMcustom: CMTable id="CmTable2" runat="server"
        Item1Text="second" Item2Text="table"
        Font-Size="Large" Font-Bold="False" Font-Italic="True"
        Alignment="Left">
    </CMcustom: CMTable>
</mobile:form>
```

ولنتمكن من تشغيل هذا التطبيق لا بد لنا من ضبط تعريف مجموعة موائمات الأجهزة الجديدة ضمن ملف الإعداد Web.config وذلك ضمن القسم <MobileControls>......<mobileControls>.

يجب تعريف مجموعة موائمات جديدة ترث من مجموعة الموائمات القياسية المعرفة ضمن الملف machine.config وإضافة إعدادات لعنصر التحكم الجديد والموائمات الخاصة به.

يمكن أن يبدوالنص المعدل للقسم <MobileControls> ضمن ملف Web.config كما يلي:

```
<configuration>
  <system.web>

    <mobileControls
        sessionStateHistorySize="6"

cookielessDataDictionaryType="System.Web.Mobile.CookielessData">
        <device name="CMcustomHtmlDeviceAdapters"
                inheritsFrom="HtmlDeviceAdapters">
           <control
name="MSPress.MobWeb.CustomControls.CMTable,CustomMobileControlLibrary"
adapter="MSPress.MobWeb.CustomControls.Adapters.HtmlCMTableAdapter,
 CustomMobileControlLibrary" />
```

```
        </device>
        <device name="CMcustomcHtmlDeviceAdapters"
                inheritsFrom="ChtmlDeviceAdapters">
            <control
name="MSPress.MobWeb.CustomControls.CMTable,CustomMobileControlLibrary"
adapter="MSPress.MobWeb.CustomControls.Adapters.HtmlCMTableAdapter,
 CustomMobileControlLibrary" />
        </device>
        <device name="CMcustomUpWmlDeviceAdapters"
                inheritsFrom="UpWmlDeviceAdapters">
            <control
name="MSPress.MobWeb.CustomControls.CMTable,CustomMobileControlLibrary"
adapter="MSPress.MobWeb.CustomControls.Adapters.WmlCMTableAdapter,
 CustomMobileControlLibrary" />
        </device>
        <device name="CMcustomWmlDeviceAdapters"
                inheritsFrom="WmlDeviceAdapters">
            <control
name="MSPress.MobWeb.CustomControls.CMTable,CustomMobileControlLibrary"
adapter="MSPress.MobWeb.CustomControls.Adapters.WmlCMTableAdapter,
 CustomMobileControlLibrary" />
        </device>
      </mobileControls>
   </system.web>
</configuration>
```

نلاحظ بأننا قمنا بمقابلة كل موائم جهاز مع عنصر التحكم بالصيغة :

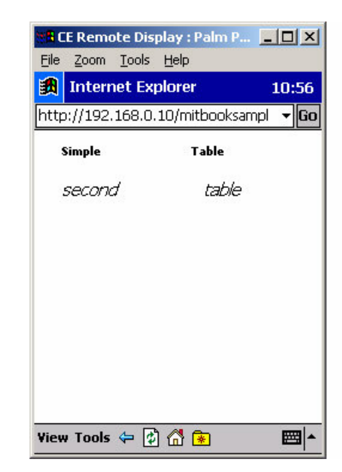
```
<control name= "controlName, assembly" adapter="adapterName, assembly"
/>
```

استخدمنا ضمن هذا النص البرمجي ملف تجميعي يحتوي صف عنصر التحكم إضافة إلى صفوف موائمات الأجهزة وهوبالاسم CustomMobileControlLibrary.dll حيث يُعرِّف كل قسم <device> مجموعة موائمات أجهزة.

لا بد لنا من الانتباه إلى إعطاء أسماء فريدة لمجموعات موائمات الأجهزة تلك.

نلاحظ أننا قد استخدمنا في مثالنا الاسماء CMcustomHtmlDeviceAdapters و CMcustomWmlDeviceAdapters و هكذا.

سنحصل عند تشغيل هذا التطبيق على نتيجة شبيهة بما يلي:





بناء عناصر التحكم المخصصة المرتبطة بالبيانات

إذا عدنا إلى عنصر التحكم الذي قمنا ببناءه عبر هذه الجلسة فسنجد أنه لن يكون مفيداً جداً إذا اقتصر على إظهار صف وحيد، ولكنه يمكن أن يصبح مفيداً إذا تم ربطه بمصدر بيانات. لذلك سنقوم بإجراء تحسينات على عنصر التحكم CMTable الذي قمنا بإنشاءه وسيكون هدفنا فهم كيفية إجاء عملية ربط عنصر تحكم مخصص بمصدر بيانات.

لدعم الربط بالبيانات يجب إجراء الخطوات التالية:

1- إضافة الخاصة من النمط ICollection والتي تحدد مصدر البيانات ويفضل تسمية هذه الخاصة باسم DataSource للمحافظة على الانسجام مع التسمية الموجودة في العناصر المرتبطة الأخرى.

2- إذا أردنا دعم الارتباط بمصادر البيانات من النمط IListSource (مثل أغراض DataSet في ASP.NET) يجب إضافة خاصة من نمط String باسم DataMember. يمكن عبر هذه الخاصة تحديد اسم عضوالبيانات أوالجدول مثلاً ليتم استخلاصه من مصدر البيانات. في مثالنا هذا سيتم دعم مصادر البيانات من النمط IEnumerable أي أنه لن يتبنى الخاصة DataMember.

3- قم بإضافة خاصة تحدد أي من عناصر البيانات سيتم استخلاصها من كل صف ضمن مصدر البيانات. في مثالنا يحتوي عنصر التحكم CMTTableDB على الخاصتين DataTextField1 و DataTextField2 التي تحدد العناصر التي سيتم إظهارها ضمن العمود الأول والثاني. كذلك يحتوي عنصر التحكم على الخاصة DataValueField التي تحدد القيمة المخفية التي يقوم التطبيق بتخزينها لكل عنصر في القائمة.

4- عندما يقوم عنصر التحكم بقراءة مصدر البيانات لا بد له من استخلاص عناصر البيانات المحددة في DataTextField1 و DataTextField2 و DataValueField وتخزينها ضمن الغرض. تشبه عناصر اللائحة ضمن عنصر تحكم CMTableDB ما هومتوفر في أغراض MobileListItem، و SelectionList و List.

5- يجب أن نقوم بتخزين أغراض CMTableListItem الذي يمثل عناصر اللائحة ضمن Collection والتي تساعد الصف CMTableDB في الوصول إليها كخاصة وليصبح بإمكاننا التعامل معها. وكما هي الحال أيضاً مع عنصر التحكم، List يمتلك عنصر التحكم المخصص CMTableDB خاصة بإسم Items للمساعدة في الوصول إلى غرض MobileListItemCollection الذي يحتوي أغراض CMTableListItem.

6- تقوم الطريقة DataBind بإداء عملية قراءة البيانات من مصدر البيانات ثم بناء أغراض CMTableListItem، وإدارج تلك الأغراض ضمن غرض MobileListItemCollection.

يمثل النص البرمجي التالي الصف CMTableListItem الذي يقوم بتخزين عناصر اللائحة. يرث هذا الصف وظائفه من الصف MobileListItem ويقوم بإضافة الخاصة Text2 لتخزين العنصر الذي يستخدمه العمود الثاني:

```
using System;
using System.Web.UI.MobileControls;

namespace MSPress.MobWeb.CustomControls
{
    /// <summary>
    /// Stores details of items displayed in the CMTableDB control.
    /// </summary>
    public class CMTableListItem : MobileListItem
    {
        // Add a property to hold text displayed in column 2.
        private String _text2;

        public String Text2
        {
            get { return _text2; }
            set { _text2 = value; }
        }

        public CMTableListItem() : base()
        {
            Text2 = "";
        }
        public CMTableListItem(
            System.Object dataItem,
            System.String text,
            System.String text2,
            System.String value)
            : base (dataItem, text, value)
        {
            Text2 = text2;
        }
    }
}
```

أما النص التالي فهوالنسخة المحسنة من الصف CMTable لدعم ربط البيانات:

```
using System;
using System.Collections;
using System.Reflection;
using System.Web.UI.MobileControls;

namespace MSPress.MobWeb.CustomControls
{
    /// <summary>
```

```
    /// Custom control built from scratch using data binding.
    /// This control inherits from PagedControl rather than
MobileControl.
    /// </summary>
    public class CMTable : PagedControl
    {
        private ICollection _dataSource = null;
        private MobileListItemCollection _items =
            new MobileListItemCollection();
        private String _title, _dataTextField1, _dataTextField2,
            _dataValueField;

        public CMTable()
        {
            Title = "";
            DataTextField1 = "";
            DataTextField2 = "";
            DataValueField = "";
        }

        public ICollection DataSource
        {
            get { return _dataSource; }
            set { _dataSource = value; }
        }
        /// <summary>
        /// Gets and sets the field displayed in the first column.
        /// </summary>
        public String DataTextField1
        {
            get { return _dataTextField1; }
            set { _dataTextField1 = value; }
        }
        /// <summary>
        /// Gets and sets the field displayed in the second column.
        /// </summary>
        public String DataTextField2
        {
            get { return _dataTextField2; }
            set { _dataTextField2 = value; }
        }

        /// <summary>
        /// Gets and sets the field stored as a hidden value.
        /// </summary>
        public String DataValueField
        {
            get { return _dataValueField; }
            set { _dataValueField = value; }
        }

        /// <summary>
```

```
        /// Gets the collection of items displayed in the table.
        /// </summary>
        public MobileListItemCollection Items
        {
            get { return _items; } }

        //InternalItemCount and ItemWeight are necessary to
        //support pagination.
        protected override int InternalItemCount
        {
            get { return Items.Count; }
        }

        // This method can be implemented in the device adapter
        // classes if the representation differs from device to device.
        // However, an item in this control always takes up one line.
        protected override Int32 ItemWeight
        {
            get { return ControlPager.DefaultWeight; }
        }

        /// <summary>
        /// Gets and sets the title displayed on some WML devices.
        /// </summary>
        public String Title
        {
            get { return _title; }
            set { _title = value; }
        }

        // Override DataBind method of base class to implement
        // data binding logic.
        public override void DataBind()
        {

            // Evaluate data binding expressions on the control itself.
            base.OnDataBinding(EventArgs.Empty);

            if (DataSource != null)
            {
                // Iterate DataSource.
                IEnumerator dataEnum = DataSource.GetEnumerator();
                while(dataEnum.MoveNext())
                {
                    // Create new item for each data item.
                    CMTableListItem item =
                        new CMTableListItem(dataEnum.Current,"","","");

                    System.Type objectType =
dataEnum.Current.GetType();
                    PropertyInfo aProp =
                        objectType.GetProperty(this.DataTextField1);
```

```
                    if (aProp != null)
                        item.Text =
                      aProp.GetValue(dataEnum.Current,null).ToString();
                    aProp =
objectType.GetProperty(this.DataTextField2);
                    if (aProp != null)
                        item.Text2 = aProp.GetValue

(dataEnum.Current,null).ToString();
                    aProp =
objectType.GetProperty(this.DataValueField);
                    if (aProp != null)
                        item.Value = aProp.GetValue

(dataEnum.Current,null).ToString();

                    // Add item to the MobileListItemCollection.
                    _items.Add(item);

                    // Add the TableListItem as a Child control
                    this.Controls.Add(item);
                }
            }
        }
    }
}
```

يجب استخدام الخاصة Item لصف عنصر التحكم CMTable ضمن طريقة Render في صف موائم الجهاز، للمرور على عناصر MobileListItemCollection والتي تقوم بتوليد تأشيرات الخرج لكل عنصر ضمن القائمة.

يكون النص التالي خاص بموائم الأجهزة الخاص بـــHTML لعنصر التحكم المخصص وهو HtmlCMTableAdapter:

```
using System;
using System.Web.UI;
using System.Web.UI.MobileControls;
using System.Web.UI.MobileControls.Adapters;
using MSPress.MobWeb.CustomControls;

namespace MSPress.MobWeb.CustomControls.Adapters
{
    /**
     * HtmlCMTableAdapter class
     */
    public class HtmlCMTableAdapter : HtmlControlAdapter
    {
        protected new CMTable Control
        {
            get { return (CMTable)base.Control; }
        }

        public override void Render(HtmlMobileTextWriter writer)
```

```
        {
            MobileListItemCollection items = Control.Items;
            if (items.Count == 0)
            {
                return;
            }

            int pageStart = Control.FirstVisibleItemIndex;
            int pageSize = Control.VisibleItemCount;
            if (items.Count < pageSize) pageSize = items.Count;
            String tableSuffix = "";
            Alignment alignment =
                (Alignment)Style[Style.AlignmentKey, true];
            if(alignment != Alignment.NotSet && alignment !=
Alignment.Left)
            {
                writer.Write("<div align=\"");
                writer.Write(alignment.ToString());
                writer.WriteLine("\">");
                tableSuffix = "\r\n</div>";
            }

            writer.AddAttribute("width","90%");
            writer.AddAttribute("cellpadding", "3");
            writer.RenderBeginTag("table");
            writer.WriteLine("");
            for (int i = 0; i < pageSize; i++)
            {
                CMTableListItem item =
                   (CMTableListItem)(items[pageStart + i]);
                writer.Write("<tr><td>");
                writer.EnterFormat(Style);
                writer.WriteEncodedText(item.Text);
                writer.ExitFormat(Style);
                writer.Write("</td><td>");
                writer.EnterFormat(Style);
                writer.WriteEncodedText(item.Text2);
                writer.ExitFormat(Style);
                writer.WriteLine("</td></tr>");
            }
            writer.RenderEndTag();
            writer.WriteLine(tableSuffix);
        }
    }
}
```

دعم التقسيم الآلي والمخصص إلى صفحات

يجب أن يدعم أي عنصر تحكم يقوم بتوليد كمية خرج كبيرة، آلية التقسيم إلى صفحات. فإذا تم استخدام مجموعة كبيرة من البيانات مع عنصر التحكم المرتبط وتم تحديد الخاصة Paginate للنموذج الحاوي على عنصر التحكم هذا إلى القيمة True، يجب السماح لعنصر التحكم بإرسال خرج مكون من مجموعات جزئية من العناصر التي يتم توزيعها على شاشات متتالية.

تتضمن عملية دعم التقسيم إلى صفحات المراحل التالية:

1- يجب أن يرث عنصر التحكم من الصف PagedControl بدلاً عن MobileControl. بالتالي سيحصل عنصر التحكم على جميع المزايا التي يمكن أن يستخدمها في التقسيم المخصص إلى صفحات: كالخاصة ItemCount والحدثLoadItems .

2- يجب القيام بإعادة تعريف الطريقة InternalItemCount لإعادة رقم العناصر الموجودة حالياً ضمن عنصر التحكم. كذلك يجب إعادة تعريف الخاصة ItemWeight لإعادة القيمة التي تحدد لإطار عمل الصفحة كم يستهلك سطر وحيد في عنصر التحكم من مساحة الإظهار على الشاشة. بعد تحديد كم من عناصر التحكم والمكونات يمكن إظهاره على شاشة الإظهار، يقوم إطار العمل بالاستعلام من كل عنصر تحكم عن خاصة ItemWeight. يقوم إطار العمل بهذا قبل تعيين كل عنصر تحكم وعنصر قائمة إلى صفحة إظهار محددة. يستهلك سطر إظهار واحد وزن 100 وحدة من وحدات النظام التي يستخدمها إطار عمل الصفحة. توجد هذه القيمة ضمن الثابت ControlPager.DefaultWeight. تعيد الخاصة ItemWeight لعنصر التحكم CMTable الذي نعمل على تطويره قيمة هذا الثابت كونه يستهلك سطر واحد لكل عنصر قائمة.

3- يقوم إطار عمل الصفحة بتعيين الخاصة Control.FirstVisibleItemIndex والخاصة Control.VisibleItemCount لتحدد العنصر الأول الذي يجب أن يظهره عنصر التحكم وعدد العناصر التي يجب إظهارها. يجب كذلك على النص البرمجي الذي نقوم بكتابته للطريقة Render ضمن صف موائم الجهاز أن يستخدم هذه المعلومات لتحديد العناصر التي سيتم إرسالها إلى الخرج.

يتم التصريح عن نموذج وب محمول لاستخدام عنصر التحكم هذا:

```
<mobile:form id="Form2" runat="server" Paginate="True">
    <CMcustom:CMTable id="CmTableDB1" runat="server" />
</mobile:form>
```

وضمن الطريقة Page_Load يتم بناء المصفوفة وعنصر التحكم المرتبط بها كما يوضح صف النص البرمجي في الخلفية:

```
public class MobileWebForm1 : System.Web.UI.MobileControls.MobilePage
{
    protected System.Web.UI.MobileControls.Form Form1;
    protected CMTable CmTableDB1;

    private void Page_Load(object sender, System.EventArgs e)
    {

        // Create large array to illustrate pagination.
        ArrayList array = new ArrayList();
        array.Add(new TeamStats("Dunes",1,38,24,8,6,80));
        array.Add(new TeamStats("Phoenix",2,38,20,10,8,70));
        array.Add(new TeamStats("Eagles",3,38,20,9,9,69));
        array.Add(new TeamStats("Zodiac",4,38,20,8,10,68));
        array.Add(new TeamStats("Arches",5,38,20,6,12,66));
```

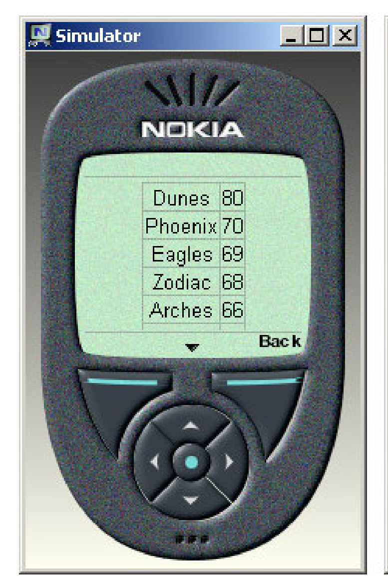
```
        array.Add(new TeamStats("Chows",6,38,17,10,11,61));
        array.Add(new TeamStats("Creation",7,38,15,12,11,57));
        array.Add(new TeamStats("Illusion",8,38,13,15,10,54));
        array.Add(new TeamStats("Torpedo",9,38,14,10,14,52));
        array.Add(new TeamStats("Generals", 10,38,14,10,14,52));
        array.Add(new TeamStats("Reaction",11,38,14,9,15,51));
        array.Add(new TeamStats("Peanuts",12,38,13,10,15,49));
        array.Add(new TeamStats("Caverns",13,38,14,6,18,48));
        array.Add(new TeamStats("Eclipse",14,38,9,15,14,42));
        array.Add(new TeamStats("Dragons", 15,38,10,12,16,42));
        array.Add(new TeamStats("Cosmos",16,38,11,9,18,42));

        CmTableDB1.DataSource = array;
        CmTableDB1.DataTextField1 = "TeamName";
        CmTableDB1.DataTextField2 = "Points";
        CmTableDB1.DataValueField = "Position";
        CmTableDB1.DataBind();
    }
    class TeamStats
    {
        private String  _teamName;
        private int _position, _played, _won, _drawn, _lost, _points;

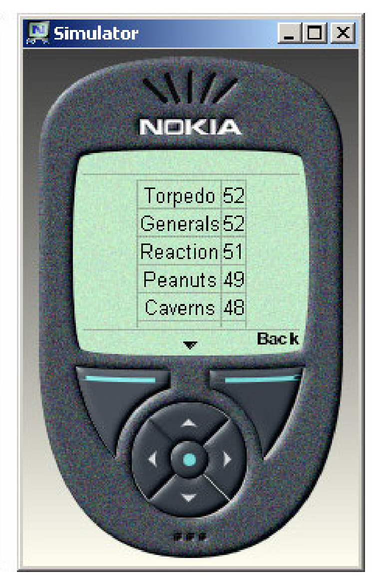
        public TeamStats(String teamName,
            int position,
            int played,
            int won,
            int drawn,
            int lost,
            int points)
        {
            this._teamName = teamName;
            this._position = position;
            this._played = played;
            this._won = won;
            this._drawn = drawn;
            this._lost = lost;
            this._points = points;
        }

        public String TeamName { get { return this._teamName; } }
        public int    Position { get { return this._position; } }
        public int    Played   { get { return this._played; } }
        public int    Won      { get { return this._won; } }
        public int    Drawn    { get { return this._drawn; } }
        public int    Lost     { get { return this._lost; } }
        public int    Points   { get { return this._points; } }
    }
}
```

يعطي هذا الصف على جهاز محمول صغير خرجاً يشبه الشكل:







استخدام الحدث OnDataBind

يمكننا إضافة أحداث عامة إلى صف عنصر التحكم المخصص.

توفر عناصر التحكم القياسية المرتبطة بالبيانات الحدث ItemDataBind والذي يمكن للمطور استخدامه لتطبيق ربط مخصص بالبيانات. لذلك من المنطقي استخدامه ضمن مثالنا الذي يتناول عنصر التحكم المخصص CMTable.

لاستخدام الحدث ItemDataBind يجب التصريح عن توكيل عام لمعالج الحدث. يمكننا التصريح عن هذا التوكيل ضمن ملف النص المصدري عند التصريح عن فضاء الاسماء ولكن خارج تعريف الصف:

```
public delegate void CMTableListItemEventHandler(
    object sender,
    CMTableListItemEventArgs e);
…..
```

يأخذ معالج الحدث معامل من النمط CMTableListItemEventsArgs كما هوحيث يتم تعريفه كما يلي :

```
public sealed class CMTableListItemEventArgs : EventArgs
{
    private CMTableListItem item;

    public CMTableListItemEventArgs(CMTableListItem item)
    {
        this.item = item;
    }
```

```
    public CMTableListItem Item
    {
        get { return item; }
    }
}
```

يمكننا بعدها تعريف التصريح عن الحدث ItemDataBind ضمن صف عنصر التحكم CMTable :

```
// Declare a static read-only object that will own the list of
registered
// event handlers
private static readonly object EventItemDataBind = new object();

public event CMTableListItemEventHandler ItemDataBind
{
    add
    {
        Events.AddHandler(EventItemDataBind, value);
    }
    remove
    {
        Events.RemoveHandler(EventItemDataBind, value);
    }
}
```

نلاحظ أن هذا الحدث يستخدم الطرائق MobileControl.Events.AddHandler و RemoveHandler.

باستخدام عنصر التحكم هذا يستطيع مطور تطبيقات الوب المحمول كتابة طرق معالجات الأحداث المخصصة لحدث ItemDataBind.يتم استخدام الطريقة OnItemDataBind لاستدعاء معالجات الأحداث التي يصرح عنها المطور:

```
protected virtual void OnItemDataBind(CMTableListItemEventArgs e)
    {
        CMTableListItemEventHandler onItemDataBindHandler =
            (CMTableListItemEventHandler)Events[EventItemDataBind];
        if (onItemDataBindHandler != null)
            onItemDataBindHandler(this, e);
    }
```

وأخيراً يمكننا استدعاء الطريقة OnItemDataBind في كل مرة يتم فيها بناء غرض CMTableListItem أثناء قراءة عنصر التحكم CMTable المعلومات من مصدر البيانات.

يجب إنشاء غرض CMTableListItemEventArgs من CMTableListItem وتمريره إلى الطريقة OnItemDataBind. بهذه الطريقة يستدعي عنصر التحكم طريقة معالج الحدث OnItemDataBind التي تم التصريح عنها، مما يوفر إمكانية تخصيص عملية الربط بالبيانات.

```
public override void DataBind()
{
    // Evaluate any data binding expressions on the control itself.
```

```
    base.OnDataBinding(EventArgs.Empty);

    if (DataSource != null)
    {
        // Iterate DataSource, creating a new item for each data item.
        IEnumerator dataEnum = DataSource.GetEnumerator();
        while(dataEnum.MoveNext())
        {
            // Create item
            CMTableListItem item =
new CMTableListItem(dataEnum.Current,"","","");

            //  intervening code not shown

            // Add item to the MobileListItemCollection of the control.
            _items.Add(item);

             // Add the TableListItem as a Child control
             this.Controls.Add(item);

        CMTableListItemEventArgs e = new CMTableListItemEventArgs
(item);
        OnItemDataBind(e);
        // After any ItemDataBind events have been called, the
        // DataItem property has no purpose and is not relevant
        // on postback, so clear it.
        item.DataItem = null;
      }
   }
```

القسم الخامس عشر:

أمان الحلول المحمولة واللاسلكية

الكلمات المفتاحية:

مفتاح، بروتكول، تشفير، خوارزمية، مفتاح خاص، مفتاح عام، مفتاح سري

ملخص:

سنتعرف في هذه الجلسة التعرف على المكونات الأساسية التي تتدخل في عملية بناء بيئة آمنة ثم ننتقل إلى التهديدات الأساسية للأمان والتي يجب الانتباه لها.

وأخيراً سوف نغطي بعض تقنيات الأمان والمقاييس التي يجب مراعاتها في تطبيقاتنا، وسنقوم بالتعرف بشكل أعمق على الأمان في البروتوكول WAP والمشاكل التي يطرحها.

أهداف تعليمية:

يتعرف الطالب في هذا الفصل على:

- مكونات البيئة الآمنة
- المخاطر الأمنية
- تقنيات الأمان
- مقاييس الأمان الأخرى
- الأمان في بروتوكول WAP
- ملاحظات بسيطة حول الأمان على مستوى التطبيق

مبادئ الأمن

قبل مناقشة عناصر الأمان المحيطة بالتطبيقات المحمولة في المؤسسات لا بد من استعراض مقدمة سريعة عن مبادئ الأمان وتقنياته. سنبدأ بالاطلاع على المكونات الأساسية التي تتدخل في عملية بناء بيئة آمنة ثم ننتقل إلى التهديدات الأساسية للأمان والتي يجب الانتباه لها، وأخيراً سوف نغطي بعض تقنيات الأمان والمقاييس التي يجب مراعاتها في تطبيقاتنا.

إنشاء بيئة آمنة:

للوصول إلى أمان (اتصال من نهاية إلى نهاية) يجب أن نأخذ بعين الاعتبار البيئة كاملة بما في ذلك طبيعة الوصول في المؤسسة، مكونات الكيان الوسيط والتطبيق الزبون.

إن ضمان أمان (اتصال من نهاية إلى نهاية) يعني ضمان أمن البيانات على كامل المسار من المرسل إلى المستقبل. وللوصول إلى هذا النوع من الأمان لا بد من تحقيق مجموعة من الأهداف الأساسية وهي:

1- التحقق من الهوية:

وهوالإجراء المتبع لإثبات كون الشخص أوالمؤسسة هي من أوما تدّعي. تتم هذه العملية في الشبكات اللاسلكية على سويتين الأولى طبقة الشبكة والثانية طبقة التطبيق.

تتطلب الشبكة التحقق من هوية المستخدم قبل الموافقة على وصوله. حيث يمكن اتمام هذه العملية ضمنياً اعتماداً على الجهاز أوالموديم أوبصورة منفصلة باستخدام تقنيات مختلفة.

أما على سوية التطبيق فتعتبر عملية التحقق من الهوية مطلوبة على سويتين الأولى سوية الزبون والثانية سوية المخدم. فللوصول إلى بيانات المؤسسة، يجب على الزبون أن يثبت للمخدم بأنه من يدّعي، وفي نفس الوقت قبل أن يسمح المستخدم لمخدم خارجي بالاتصال به يجب على المخدم أن يظهر هويته إلى التطبيق الزبون.

إن أبسط طرق التحقق من الصحة هي استخدام اسم المستخدم وكلمة المرور أما الطرق الأكثر أماناً وتعقيداً فتتضمن الشهادات الرقمية والتوقيع الرقمي.

2- سلامة البيانات:

إن التحقق من سلامة البيانات يعني التأكد بأن البيانات لم يتم تعديلها أواعتراضها بأي طريقة أثناء نقلها بين المرسل والمستقبل. يمكن الوصول إلى هذا الغرض باستخدام تشفير البيانات مع تقنية اختبار المجموع CheckSum أورمز رسالة التحقق MAC حيث يتم تشفير هذه المعلومات ضمن الرسالة بتطبيق خوارزمية خاصة عليها. وعند استقبال هذه الرسالة من قبل المستقبل يتم حساب MAC ومقارنته مع الـــ MAC المُشفَّر ضمن الرسالة للتأكد من تطابقهما. إذا تم التطابق يستطيع المستقبل التأكد من أنه لم يتم العبث بالرسالة وإلا فيمكن للمستقبل إهمال الرسالة لكون معلوماتها غير صحيحة.

3- السرية:

السرية هي أحد أهم نواحي الأمان. وهي تتلخص في المحافظة على خصوصية البيانات والتأكد بأنه لا يمكن رؤيتها من قبل أطراف غير مرغوب بهم.

عندما يتكلم الأشخاص عن الأمان في النظام فهم غالباً ما يهتمون بالمعلومات الحساسة كمعلومات بطاقات الإئتمان والسجلات الصحية، التي لا يجب أن تكون مرئية للأشخاص ذوي النوايا السيئة. تُعتبر عملية تشفير البيانات بحيث تصبح غير مقروءة لأي شخص عدا المُعتَرَف عليه هي الطريقة الأكثر شيوعاً في هذا المجال.

4- السماحية:

هي عبارة عن إجراء تحديد سوية الوصول لمستخدم ما وحق هذا المستخدم بالقيام بأفعال معينة. تكون السماحية عادةً شديدة الالتصاق بموضوع التحقق من الهوية، فما أن يتم التحقق من هوية مستخدم حتى يتم تزويده بالسماحيات المحددة له. يجري عادةً استخدام ما يسمى لوائح التحكم بالوصول ACL للمساعدة في تحديد السماحيات.

فعلى سبيل المثال، قد يمتلك جميع المستخدمين حق القراءة من مصدر البيانات ولكن يمكن فقط لمدير النظام ومصادر أخرى موثوقة أن تمتلك حق الوصول والكتابة فيه.

عدم الإنكار:

يتلخص عدم الإنكار في جعل الأطراف مسؤولة عن المناقلات التي اشتركت أوقامت بها. تتضمن هذه العملية التعرف على الأطراف بطريقة لا يمكنهم فيها لاحقاً إنكار اشتراكهم في مناقلة ما مما يعني أنه بإمكان مرسل الرسالة أومستقبلها التأكيد لطرف ثالث بأن المرسل قد أرسل هذه الرسالة فعلاً، وبأن المستقبل قد قام باستلام نفس هذه الرسالة. للقيام بهذا العمل لا بد من استخدام توقيع الكتروني وختم زمني يمكن التحقق منه مع كل رسالة من قبل طرف ثالث موثوق.

المخاطر الأمنية

تُعتبر عملية بناء حل أمان عملية صعبة لايمكن تنفيذها دون التعرف وبصورة واضحة على المخاطر المحتملة.

فسواء كانت البيانات تنتقل بصورة سلكية أولاسلكية، لا بد من اتخاذ الاحتياطات اللازمة ضد المخاطر التالية:

انتحال الهوية: هوعبارة عن عملية الحصول على وصول إلى نظام أوتطبيق ما بالتظاهر بأنه مستخدم ما. فإذا تمكن المستخدم من الوصول إلى النظام أصبح بإمكانه إنشاء استجابات مزورة أورسائل قد يكون غرضها الاستعلام عن معلومات إضافية للحصول على سماحية أعلى ضمن النظام. لذا يُتبر انتحال الهوية مشكلة أساسية في الأمن على الإنترنت لأنه يعطي للشخص المهاجم إمكانية إيهام مستخدمي التطبيق بأنهم يتصلون بمصدر موثوق كالبنك مثلاً، بينما هم يتصلون حقيقة مع جهاز المهاجم مما يساعد المهاجم في الحصول على معلومات هامة وحساسة.

التنصت:

هي عبارة عن آلية تستخدم لمراقبة تدفق البيانات عبر الشبكة. يمكن لهذه الآلية أن تُستخدم لأغراض جيدة. ترتبط هذه العملية عامة بعملية نسخ غير مسموحة للبيانات على الشبكة بحيث يتم الاستماع لكل ما يدور عليها وبالتالي يمكن لأطراف غير مرخصين الحصول على معلومات حساسة تسمح لهم بالتسبب بأذى لنظام المؤسسة أولمستخدمي التطبيق.

تعتبر عملية التنصت خطرة لأنها سهلة، وبات الحصول على الأدوات الخاصة بها سهلاً.
يمكن الحل في عملية تشفير البيانات والتأكد من أن خوارزمية التشفير غير قابلة للفك (قدر الإمكان) فعلى سبيل المثال، اكتشف الكثير من مستخدمي الشبكات اللاسلكية بأن التشفير WEP ليس كافياً لحماية البيانات.

العبث:

العبث بالبيانات (ويسمى أيضاً تهديد سلامة البيانات) هوعملية التعديل على البيانات من الحالة الأصلية لها. غالباً ما تتضمن هذه العملية اعتراض إرسال البيانات كما يمكن أن تحدث على البيانات المخزنة على المخدم أوالزبون.
يتم تمرير البيانات عادة على أنها البيانات الأصلية.
تعتبر عمليات التحقق من الهوية والتحقق من السماحية من وسائل الحد من هذا التهديد.

السرقة:

تعتبر السرقة مشكلة متأصلة في الحلول المحمولة، إذ لا نتكلم فقط عن سرقة الجهاز، بل عن سرقة أي معلومات قد تكون عليه.
يمكن أن تشكل هذه المشكلة تهديداً كبيراً في حالة التطبيقات العاملة على أجهزة الزبون الذكي كونها تحتوي على البيانات التي غالباً ما تكون سرية، لذلك يجب اللجوء إلى القواعد التالية لتأمين التجهيزات المحمولة:

1- إقفال الجهاز بتركيبة من اسم مستخدم وكلمة مرور لمنع الوصول السهل إلى البيانات.
2- التحقق من الهوية عن الدخول إلى أي تطبيق على الجهاز.
3- عدم تخزين كلمات المرور على الجهاز.
4- تشفير المعلومات على أي واسطة تخزين.
5- تحسين سياسات الأمان العامة لمستخدمي الأجهزة المحمولة.

تقنيات الأمان

بعد أن قمنا بالاطلاع على المخاطر التي يمكن أن تهدد الأمان في الحلول المحمولة سنستعرض أهم التقنيات المستخدمة للتقليل من هذه المخاطر :

تعمية وإخفاء البيانات:

تهدف تعمية وإخفاء البيانات إلى السماح لطرفين بالاتصال وتبادل البيانات بشكل يمنع طرف ثالث من فهم ما يجري إرساله.
تعتبر هذه التقنية أحد المتطلبات الأساسية في بيئة آمنة كونها تتعامل مع جميع نواحي النقل الآمن للبيانات بما يتضمن التحقق من الهوية والتوقيع الالكتروني.
تُعتبر مبادئ عملية التعمية بسيطة، ولكنها عملية معقدة جداً من الناحية العملية وبالأخص في حالة التطبيقات المحمولة الكبيرة.

الخوارزميات والبروتوكولات والتطبيقات:

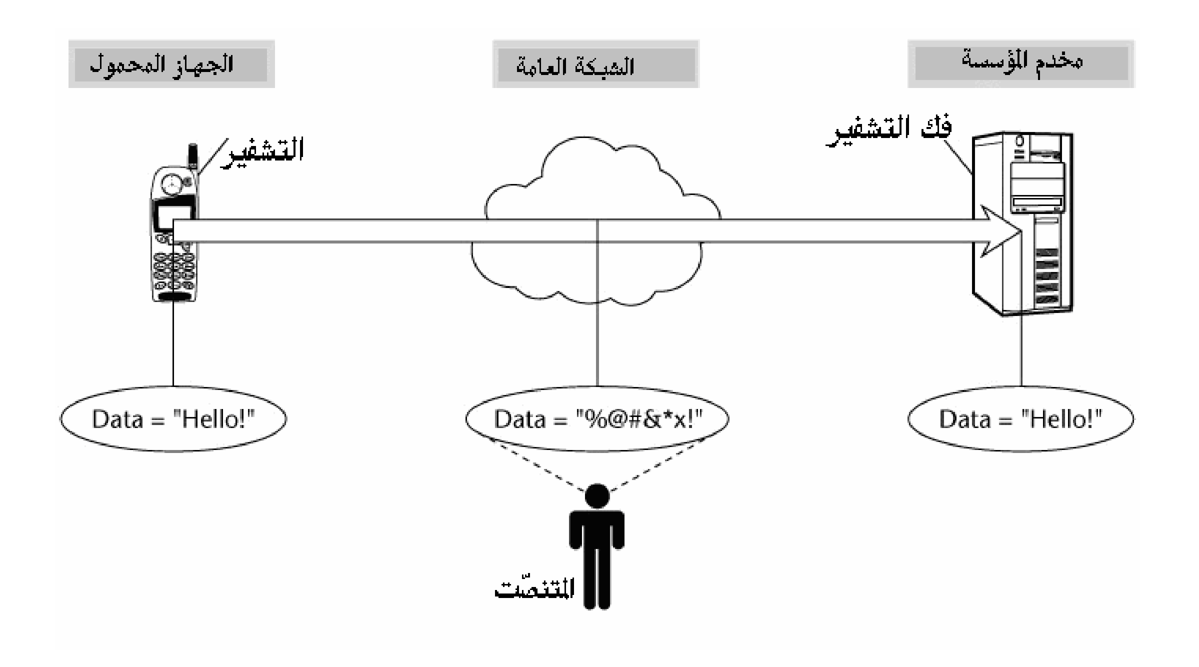
تعمل التعمية في عدة سويات. السوية الدنيا هي سوية الخوارزميات. تصف هذه الخوارزميات المراحل المطلوبة لعملية تتمحور حول تحويل البيانات من تنسيق إلى آخر. أما البروتوكول فيعبر عن التوصيف الكامل للإجرائية التي تقوم بتنفيذ فعل التشفير بما يتضمن

المعلومات اللازمة حول أي طارئ يمكن أن يحصل أثناء هذه العملية.
إن التمييز بين خوارزمية التشفير والبروتوكول هام جداً لأن استخدام خوارزمية تشفير ممتازة لا يعني بالضرورة ترجمتها إلى بروتوكول جيد، فالبروتوكول مسؤول عن أكثر من عملية التشفير بحد ذاتها ويتضمن عمله نقل البيانات إضافة إلى تبادل المفاتيح.
أما التطبيقات فتتوضع فوق سوية البروتوكول، ولايعني البروتوكول الجيد بالضرورة تطبيقاً آمناً لأن التطبيق بحد ذاته قد يقود إلى نقاط ضعف ومشاكل أمن أخرى.

تشفير البيانات:

يُعتبر التشفير قلب أي عملية إخفاء للبيانات هوإجراء تحويل مجموعة بيانات تسمى النص الصريح إلى نمط غير قابل للقراءة يسمى النص المشفر مما يسمح بالحفاظ على خصوصية البيانات الحساسة حتى عند تمكن أشخاص غير مرخصين من الوصول إليها.
الطريقة الوحيدة للتمكن من قراءة البيانات هي بتحويلها مجدداً إلى النموذج الأصلي باستخدام إجراء يسمى فك التشفير. تدعى طريقة التشفير وفك التشفير الخوارزمية أوالشيفرة.

يوضح الشكل التالي مبدأ التشفير، حيث يتم نقل المعلومات عبر قناة نقل عامة بشكل مشفر لمنع أي متلصص على الخط من فهم البيانات التي يتم إرسالها.



تستخدم الخوارزميات الحديثة المفاتيح للتحكم بعملية تشفيروفك تشفير البيانات. فعند تشفير رسالة ما لن يتمكن إلا المستخدم الذي يحمل المفتاح المناسب من فك تشفير هذه الرسالة. تصنف خوارزميات التشفير المبنية على المفتاح إلى نوعين: المتناظرة وغير المتناظرة.

تستخدم الخوارزميات المتناظرة مفتاح وحيد لتشفير وفك تشفير كل الرسائل. يستخدم المرسل المفتاح ليشفر الرسالة ومن ثم يرسل الرسالة إلى المستقبل المطلوب. عند استلام الرسالة يستخدم المستقبل نفس المفتاح لفك تشفير الرسالة.
تعمل هذه الخوارزمية بشكل جيد عند إيجاد طريقة آمنة لنقل مفتاح التشفير بين المستخدمين، كالاجتماع مثلاً قبل عملية نقل

البيانات. للأسف تظهر مشكلة كبيرة عند الاضطرار إلى تبادل المعطيات بين أطراف غير مرتبطين بشكل مباشر. مثل حالة موقع تجارة الكترونية وزبون يحاول الشراء من هذا الموقع حيث تصبح عملية تبادل مفتاح التشفير هومشكلة بحد ذاته. تسمى تقنية التشفير المتناظر أيضاً تشفير المفتاح السري.

أكثر النماذج شيوعاً لهذه الطريقة هي التشفير القياسي للبيانات DES. وقد تم بعدها تطوير العديد من التقنيات الأكثر أماناً منها التشفير القياسي المتقدم AES المبني على خوارزمية Rijndael كذلك تقنية 3DES وخوارزمية تشفير البيانات العالمية IDEA و Blowfish ومجموعة خوارزميات Rivest و RC2,RC4,RC5 و RC6.

التشفير غير المتماثل يتناول المشكلة الأساسية التي تشوب التشفير التماثلي وهي استخدام مفتاح وحيد. في عام 1975 قام عالمان هما Deffie و Helman بتطوير حل باستخدام مفتاحين مختلفين يستخدم الأول في عملية التشفير والثاني في فك التشفير. يسمى المفتاح المستخدم في التشفير المفتاح العام. يتم توزيع هذا المفتاح بصورة عامة عبر قنوات غير آمنة للاستخدام العام. أما المفتاح المستخدم لفك التشفير فيسمى المفتاح الخاص. لا يتم نقل هذا المفتاح لأنه مطلوب فقط من قبل الطرف الذي يقوم بفك تشفير البيانات. تكون هذه المفاتيح متصلة بشكل ضمني ومبنية على أعداد أولية كبيرة وتوابع وحيدة الاتجاه. تجعل هذه التقنية من غير المجدي رياضياً حساب المفتاح الخاص من المفتاح العام.

يمكن اختراق بعض أنظمة التشفير المنية على مفاتيح بطول 64بت مثل DES باستخدام التجريب بالقوة وذلك بمحاولة اختبار جميع الاحتمالات الممكنة لحين إيجاد التركيبة المناسبة.

تقنيات الأمان: الشهادات الرقمية

توفر الشهادات الرقمية طريقة لضمان كون المفتاح العام ينتمي إلى الطرف الذي يمثله. ولكي يتحقق هذا الأمر يجب أن يتم التحقق أيضاً من الشهادة للتأكد من أنها تمثل الجهة المدعية (شخص أومؤسسة).

تتم هذه العملية بالاعتماد على طرف ثالث يسمى سلطة منح الشهادات Certificate Authority. من أهم سلطات منح الشهادات عالمياً شركات مثل VeriSign، Entrust، Certicom.

تحتوي الشهادات الرقمية عادة على التالي:

- اسم حامل الشهادة والمعلومات المميزة لهذا الشخص.قد تتضمن معلومات إضافية أيضاً عن محدد المصدر القياسي لمخدم الوب الذي يستخدم هذه الشهادة وعنوان البريد الخاص به.
- المفتاح العام المخصص لحامل الشهادة.
- اسم سلطة منح الشهادات التي أصدرت هذه الشهادة.
- مدة صلاحية الشهادة (عادة تاريخ البداية والنهاية).
- توقيع رقمي من سلطة منح الشهادات للتحقق من أنه لم تتم عملية عبث بالمناقلة.

كذلك يملك المستخدمون خيار توقيع شهادة رقمية بنفسهم بحيث يكونون سلطة مانحة للشهادات لأنفسهم. هذا الطرف الإضافي يمكن أن يعتبر موثوقاً إذا كان هوالآخر موقعاً بمفتاح موثوق. بهذه الطريقة يمكن استمرار متابعة الشهادة للوصول إلى السلطة الموثوقة الأساسية

التي قامت بمنح هذه الشهادة وإلا فتعتبر كامل السلسلة غير موثوقة.

من أهم التنسيقات المعيارية لهذه الشهادات التنسيق X.509 حيث تعتبر هذه الشهادات مشهورة ضمن تطبيقات الانترنت. أما ضمن مجال التقنيات اللاسلكية فيستخدم نوع آخر من الشهادات الرقمية يسمى WAP Server WTLS Certificate حيث تعتبر هذه الشهادات شهادات مختصرة عن شهادات WTLS (التي سنأتي على ذكرها لاحقاً ضمن هذه الجلسة) وهي نسخة مبسطة عن المعيار X.509 تم إنشاؤها لأن المعيار شهادات X.509 كبيرة لاتتناسب مع طبيعة التطبيقات اللاسلكية.تستخدم شهادات WTLS عادة في تطبيقات WAPعندما يريد المستعرض الصغري التحقق من هوية مخدم WAP وتشفير البيانات باستخدام WTLS.

تقنيات الأمان: التوقيع الرقمي

يُستخدم التوقيع الرقمي للتحقق من كون الرسالة قد أتت بالفعل من المرسل المدعي. يكون هذا التوقيع مبني على مبدأ أن من أنشئ التوقيع يملك المفتاح الخاص وبحيث يمكن التحقق منه باستخدام المفتاح العام.

يتم إنشاء التوقيع الرقمي بتلخيص الوثيقة ثم دمجها مع معلومات عنالموقّع وختم زمني ومعلومات أخرى مطلوبة.
عملية التلخيص هي عبارة عن تابع يقوم بأخذ حجم كيفي من الوثيقة (الرسالة) وتوليد خرج بحجم ثابت يدعى الطبعة. يتم بعدها تشفير هذه المعلومات باستخدام مفتاح خاص من قبل المرسل باستخدام خوارزمية غير متناظرة مناسبة. تسمى الكتلة المشفرة الناتجة التوقيع اللكتروني.

تكون طبعة الرسالة الذي تم حسابها عبارة عن قيمة تمثل الحالة الراهنة للوثيقة. فإذا تم تعديل الوثيقة أوالعبث بها بأي طريقة ستتغير قيمة هذه الطبعة.

بدمج هذه الطبعة في بنية التوقيع الالكتروني يصبح بإمكان المستقبل اكتشاف أي عملية عبث بالوثيقة منذ إنشاء التوقيع الالكتروني.

البنية التحتية للمفتاح العام:

البنية التحتية للمفتاح العام أوما يسمى ب PKI هوالتعبير الذي يصف منظمة كاملة من النظم والقواعد التي تُعرِّف نظام أمان وحيد.

تُعرِّف مجموعة عمل IETF X.509 البنية التحتية للمفتاح العام على أنها مجموعة الكيانات الصلبة، والبرمجيات، والأشخاص، والإجراءات اللازمة لإنشاء، وإدارة، وتخزين، وتوزيع، ورفض الشهادات المبنية على تعمية المفتاح العام.

تتضمن PKI المكونات التالية:

- سلطات مانحة للشهادات مسؤولة عن عملية إصدار أورفض الشهادات.
- سلطات تسجيل مسؤولة عن ربط المفاتيح العامة مع هوية حامليها.
- حاملوالشهادات الذين تم إصدار شهادات لهم والتي بإمكانهم استخدامها لتوقيع الوثائق الالكترونية.

- موضع تخزين لتخزين معلومات الشهادات الصادرة وتلك التي تم إبطالها أورفضها.
- سياسة أمان تحدد الأمان كتوجه أساسي للمنظمة.

البرتوكولات الرئيسية المستخدمة في النقل الآمن للبيانات:

برتوكول طبقة المقبس الآمن SSL (Secure Socket Layer):

يعتبر هذا البروتوكول من البروتوكولات الهامة المستخدمة حالياً على الانترنت. تم تطويره من قبل شركة Netscape لتأمين جلسات آمنة خاصة وهويُستخدم بصورة أساسية فوق البروتوكول HTTP علماً أنه يمكن أن يستخدم فوق بروتوكول أخرى منها FTP.

يُستخدم هذا البروتوكول مزيج من الخوارزميات المتناظرة وغير المتناظرة لرفع الأداء.

هناك أربع مراحل أساسية ضمن جلسة SSL هي:

1- المصافحة والتفاوض على الشيفرة: حيث يتفق كل من الزبون والمخدم على الخوارزميات أوالشيفرة المستخدمة.
2- التحقق من الهوية: حيث يتم التحقق من هوية المخدم (وبشكل كيفي الزبون) باستخدام شهادة رقمية.
3- تبادل المفتاح: يقوم الزبون بإنشاء مفتاح سري ويرسله إلى المخدم باستخدام تشفير المفتاح العام. يقوم المخدم بفك تشفير الرسالة باستعمال المفتاح الخاص. بعد هذا سيتمكن الزبون والمخدم بالاتصال حتى نهاية الجلسة وإرسال المعلومات مشفرة باستخدام المفتاح السري.
4- تبادل بيانات التطبيق: ما أن يتم تأسيس جلسة آمنة تماثلياً يمكن أن يتم تناقل البيانات المشفرة بين الزبون والمخدم.

يمكن استعمال SSL من قبل العديد من الأجهزة المحمولة.
يمكنك تحديد كونك تستعمل SSL أم لا من ظهور اسم البرتوكول https عوضاً عن http في بداية محدد المصدر القياسي.
إن تشفير البيانات ضمن البرتوكول أكثر فعالية من تشفير البيانات فقط وإرسالها باستخدام HTTP لأن عملية التشفير ضمن البرتوكول تقوم بتشفير البيانات على مستوى الحزمة أما إذا قمت بتشفير البيانات ثم إرسالها باستخدام البرتوكول فلن تستتطيع فك التشفير لحين وصول جميع الحزم الخاصة بالرسالة المرسلة.

بروتكول أمان طبقة النقل TLS (Transport Layer Security):

يعد TLS الجيل الجديد من برتوكول SSL حيث يتكون من طبقتين: الطبقة الأدنى هي طبقة بروتوكول السجل Record protocol والذي يتوضع عادة فوق طبقة بروتوكول نقل ذواعتمادية عالية كبروتوكول TCP.
أما الطبقة العليا من بروتوكول TLS فهي طبقة بروتوكول المصافحة Handshake. توفر هذه الطبقة الاتصال الآمن باستخدام التحقق من الهوية الذي يستعمل التعمية المتناظرة، والتفاوض على المفتاح السري، وتزويد تفاوض ذواعتمادية عالية.
كما هي الحال في SSL فـــ TSL مستقلة عن الشيفرة أي تستطيع استخدام طيف من الشيفرات.
تكمن الأهداف الأساسية وراء استخدام TLS في أمان التشفير، وقابلية التشغيل البيني، وإمكانية التوسع.

برتوكول أمان طبقة النقل اللاسلكي WTLS:

WTLS هي طبقة الأمان المعرفة ضمن المعيار WAP. ويعمل فوق البروتوكول Transport Protocol Layer مما يجعله مناسباً للعديد من البروتوكولات اللاسلكية الأخرى.
يشابه هذا البروتوكول، البروتوكول TLS ولكن تم تحسينه للعمل بشكل أفضل على الشبكات ذات زمن التأخير العالي وعرض الحزمة الضيق. كذلك يضيف هذا البروتوكول ميزات دعم حزم البيانات وأمثلية عملية المصافحة وتحديث المفتاح. كذلك يدعم استخدام شهادات WTLS للتحقق من الهوية من طرف المخدم.

بروتوكول IP Security (IPSec):

يختلف بروتوكول IPSec عن البروتوكولات الباقية في أنه لا يعمل في طبقة التطبيق وفي حين تعمل بروتوكولات SSL، TLS، WTLS لتأمين اتصال آمن على شبكة غير آمنة يتوجه IPSec إلى جعل الانترنت بحد ذاتها آمنة.
يساعد IPSec هذا في توفير خدمات تحقق من الهوية، سلامة البيانات، والخصوصية على طبقة حزمة البيانات.
استهدف IPSec بصورة أساسية من بين الأجهزة المحمولة زبائن أجهزة الحاسب المحمول. ولكن بدأت وبشكل سريع أنواع الأجهزة المحمولة المختلفة تدعم الشبكات الافتراضية الخاصة المبنية على IPSec.
سيزداد انتشار استخدام هذا البرتوكول مع دعم برتوكول IP6 الذي يضمن IPSec كجزء من المعيار الخاص به. من المهم أيضاً معرفة أن IPSec يدعم TCP/IP ولا يدعم WAP.

مقاييس أمان أخرى

يجب الأخذ بعين الاعتبار أيضاً عند تطبيق الحل المحمول العديد من مقاييس الأمان الأخرى التي تستخدم كتقنيات لرفع سوية الأمن العام للنظام. أهم هذه التقنيات هي:

الجدران النارية :
تشكل هذه الجدران حاجزاً في الشبكة بين ما هوخاص وعام. وجدار النار عبارة عن مجموعة من البرمجيات التي عادة ما يتم توضيعها على مخدم عبّارة منفصل تكون وظيفته الأساسية تقييد الوصول إلى مصادر الشبكة الخاصة من المستخدمين في الشبكات الأخرى. عند تثبيت إمكانية الوصول إلى المؤسسة عبر الانترنت سيكون من الضروري تركيب جدار ناري لحماية المصادر ضمن المؤسسة وأحيانا للتحكم بالمصادر التي يستطيع مستخدموشبكتنا الوصول إليها.بالنسبة للتجهيزات المحمولة التي تمتلك اتصال دائم يشكل وجود جدار نار شخصي على الجهاز المحمول فكرة جيدة.

الشبكات الافتراضية الخاصة VPN:
تسمح شبكات VPN بتحويل شبكة عامة (عادة الانترنت) إلى شبكة خاصة. تمكن هذه التقنية الأشخاص الذين يعملون عن بعد من الاتصال بشبكة المؤسسة بطريقة آمنة عبر الانترنت.
قبل وجود هذة التقنية كان لا بد لهؤلاء الأشخاص الحصول على خطوط خاصة مؤجرة للوصول إلى نفس النتيجة. تستخدم تقنية VPN

لتجاوز مشاكل الأمن في شبكات WLAN.

التحقق من الهوية المبني على معاملين:

عادة عند التعامل مع المناقلات المالية أوذات الطابع الحساس تكون الحاجة ماسة إلى آلية تحقق من الهوية على السوية المناسبة.تتلخص هذه الطريقة في حاجة المستخدمين إلى المرور في عملية تحقق مزدوج من الهوية عادة ما يكون باستخدام كلمة مرور مثلاً إضافة إلى امتلاك المستخدم لبطاقة تقوم بتوليد كلمة سر لمرة واحدة عند كل دخول. دمج هذين المعاملين يجعل من الصعب جداً على الأشخاص غير المرخصين بالوصول إلى النظام.

المقاييس الحيوية:

حتى باستخدام التحقق من الهوية المبني على عاملين يمكن للمخترقين الوصول إلى النظام.

للتخلص من هذا يمكن استبدال كلمة السر بنموذج أقوى من التحقق من الهوية وهوذلك المبني على المقاييس الحيوية مثل تلك التي تعتمد البصمة، التعرف على شكل الوجه، شكل قزحية العين، التعرف على الصوت...إلخ مشكلة هذه الطرق أن اعتماديتها غير عالية ففي الكثير من الأحيان تؤدي إلى ما يسمى الرفض الخاطئ وعدم السماح لأشخاص مرخصين بالدخول.

سياسة الأمان:

أخيراً وليس آخراً لا بد من اعتماد سياسة أمان ضمن الشركة أوالمؤسسة. إن مقاييس الأمان المتبعة التي استعرضنا بعضها هي انعكاس لسياسة الأمان التي يجب أن تقوم بوضع النقاط الرئيسية لكل نواحي الأمان في المؤسسة بما يتضمن التقنيات وكيفية استخدامها. حتى في حال اعتماد المؤسسة حلول أمنية متفوقة تقنياً سيظل النظام غير آمن إذا لم يتبع المستخدمون تعليمات الأمان الضرورية.

يجب دائماً أن نتذكر بأن المتطفلين يهاجمون النقطة الأضعف في النظام والتي أحياناً لا تكون الجانب التقني ولكن المستخدمون أنفسهم.

أمان بروتوكول WAP

لطالما تعرض بروتوكول WAP للانتقاد بسبب ضعف تغطيته للنواحي الأمنية. سنتعرف في هذا الجزء من الجلسة على المشاكل الأمنية في برتوكول WAP وكيفية تجاوزها.

لا بد من تغطية النقاط التالية في بنيان الأمان الخاص بالنسخة WAP1.x:

TLS:

أمان طبقة النقل والمعروف بالقناة الآمنة. يقوم بالتعامل مع الاتصال نقطة إلى نقطة بين الزبون اللاسلكي ومصدر البيانات في المؤسسة. تتخلل هذه العملية عملية الاتصال عبر قنوات لاسلكية وسلكية. في WAP يتم تشفير البيانات أثناء نقلها عبر الأثير باستخدام بروتوكول WTLS وعبر الأسلاك باستخدام بروتوكولات مثل SSL أوTLS. يقود هذا الاختلاف إلى أحد المشاكل الأساسية في أمان WAP. قبل الدخول في تفصيل المشكلة سنتعرف بشكل أوسع على البروتوكول WTLS وميزاته.

تم تطوير بروتوكول WTLS ليغطي حاجة بيئة الشبكات اللاسلكية بمواصفاتها الخاصة كضيق عرض الحزمة، والتأخير العالي على الشبكة لتكون البديل اللاسلكي لبروتوكول TLS. والذي لا يمكن استخدامه بفعالية في البيئة اللاسلكية.

تمثل النقاط التالية أهم المميزات التي تمت إضافتها إلى برتوكول WTLS والتي لا يقدمها TLS:

- دعم خوارزميات تعمية مختلفة. في حين تدعم SSL وTLS تشفير RSA تدعم WTLS تشفير RSA وDH وECC.
- تعريف شهادة مفتاح عام مدمج وشهادات WTLS وهي أكثر فعالية من نسخة شهادات X509.
- دعم حزم بيانات UDP. يتناول هذا عدة مناطق من البروتوكول ابتداءاً من تشفير البيانات وحتى إدارة الرسائل، تكرارها وترتيبها.
- خيار تحديث المفتاح المستخدم حيث يتم التفاوض بتباعد زمني على تحديث المفتاح المستخدم وذلك تبعاً لعدد الرسائل المرسلة.
- مجموعة أوسع من التحذيرات بما يضيف وضوح أعلى عند معالجة الا×طاء.
- تحقيق أمثلية المصافحة مما يقلل من عدد الدورات المطلوبة لتحقيق هذه العملية بشكل يناسب البيئة ذات التأخير العالي للشبكات اللاسلكية.

بالإضافة إلى هذه الميزات تقدم WTLS ثلاث سويات من التحقق من الصحة بين الزبون والعبّارة هي التالية:

Class I WTLS: تفاعل بين الزبون وعبارة WAP بدون تحقق من الهوية.

Class II WTLS : يقوم المخدم بإظهار هويته للزبون باستخدام شهادة WTLS.

Class III WTLS: يقوم كل من عبارة WAP والزبون بالتحقق من هوية الطرف الآخر.

يستخدم هذا النوع من التحقق من الهوية باستخدام البطاقات الذكية.فمثلاً يمكن لــ (SIM) في نظام GSM تخزين معلومات التحقق من الصحة.

الثغرة في برتوكول WAP

لسوء الحظ وبالرغم من تقديم WTLS للعديد من المميزات التي تضاهي برتوكول TLS بالنسبة للنقل الآمن للمعلومات لاسلكياً إلا أنه تسبب بمشكلة أساسية.

تتلخص المشكلة في ضرورة استخدام WTLS وTLS ضمن نفس البنيان الخاص بـــ WAP. ففي مرحلة معينة هناك حاجة للترجمة بين البرتوكولين. من هذه النقطة وليس من برتوكول WTLS نفسه تبرز مشكلة ثغرة أمنية.

إن عملية الترجمة تلك تتم على عبّارة WAP حيث يتم استخدام WTLS بين الجهاز المحمول والعبّارة وبروتوكول TLS بين العبّارة ومخدم المؤسسة. في هذه المرحلة وضمن عبّارة WAP تتم عمل فك تشفير للمحتوى المُشفر باستخدام WTLS وإعادة تشفيره باستخدام البروتوكول TLS وأثناء عملية فك وإعادة التشفير تلك تتواجد المعلومات بصيغة نص صريح. مع أن الزمن الذي تبقى فيه المعلومات بصيغتها الصريحة أصغري ومع أن العبارة غير موجودة ضمن النطاق العام ولكن تظل هذه الثغرة نقطة لايُستهان بها في موضوع الأمان ضمن المعيار WAP.

هناك خياران للتعامل مع ثغرة WAP:

- قبول وجود نقطة الضعف تلك على العبّارة وبذل كل الجهود لحماية العبّارة باستخدام الجدران النارية، وآليات المراقبة وتشديد سياسة الأمان ما أمكن.
- نقل عبارة WAP إلى داخل نطاق المؤسسة وإدارتها ذاتياً.

الاختيار بين هذين الحلين هوعبارة عن قرار يرتبط بالمؤسسة وهوعبارة عن مقايضة بين استهلاك مصادر إضافية لإدارة تطبيق عبارة WAP ضمن المؤسسة، والتهديدات الأمنية المحتملة على بيانات المؤسسة. لحسن الحظ تم إيجاد حل لهذه المشكلة ضمن المعيار WAP2.x

WAP2.x:

هناك العديد من الميزات الجديدة التي حملها المعيار WAP2.x ولكن كان الأكثر أهمية منها دعم بروتوكولات الانترنت القياسية. قاد هذا إلى استخدام برتوكولات مثل HTTP، TCP، IP وتمكين استخدام البرتوكول TLS للنقل الآمن للبيانات.

نظراً لزوال الحاجة لاستخدام WTLS أصبح بإمكان بروتوكول واحد هوTLS أن يستخدم من الجهاز الزبون إلى مخدم المؤسسة وبالتالي الحصول على اتصال من نهاية إلى نهاية حقيقي وآمن والتخلص ثغرة WAP تماماً.

الأمان على مستوى التطبيق

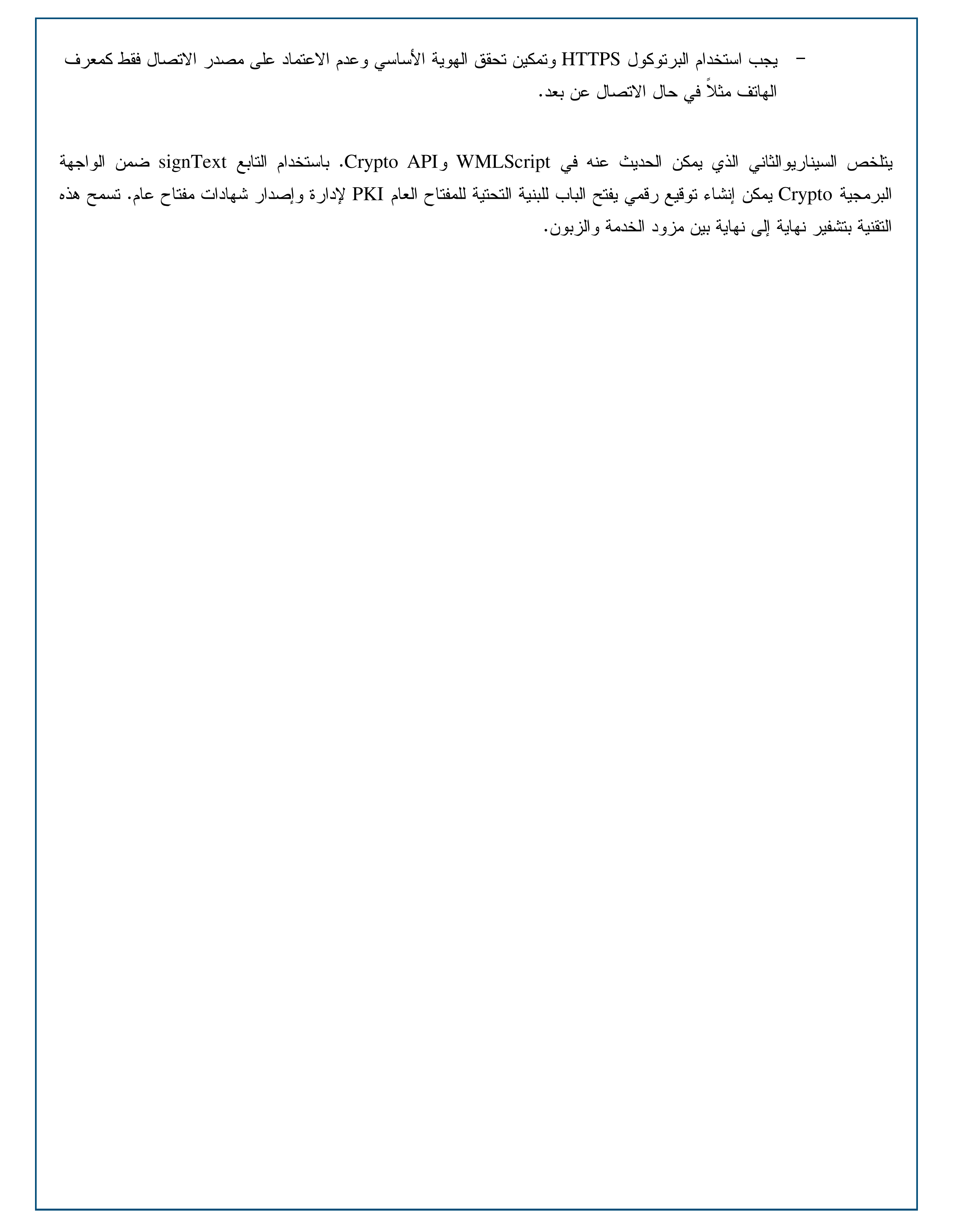
مع كل الانتباه إلى ثغرة WAP وبروتوكول TLS ينسى المطورون غالباً الأمان على مستوى التطبيق.

يعتبر الأمان على مستوى التطبيق ضروري في موضعين:

1- عندما يكون الأمان مطلوباً بما يتجاوز نقاط النهاية لبرتوكول TLS.
2- عندما يكون المطلوب الوصول إلى المحتوى التقديمي وليس إلى بيانات المؤسسة.

يتلخص السيناريوالأول الذي يمكن الحديث عنه في استخدام تقنيات تزودها WML. بشكل عام، يتم تعيين لإعدادات الافتراضية إلى أعلى درجة أمان. فيما يلي بعض النقاط الواجب ملاحظتها:

- يجب على كل بطاقة WML تطلب الوصول إلى معلومات حساسة إسناد القيمة True إلى الواصفة sendreferer ضمن التأشيرة <go>.
- يجب على النص البرمجي الذي يقوم بمعالجة الطلبات الخاصة بالمعلومات الحساسة التأكد من محدد المصدر القياسي المعين في الترويسة REFERER ضمن طلب HTTP للتأكد من أن الطلب الذي تتم معالجته قادم من نطاق صديق.



- يجب استخدام البرتوكول HTTPS وتمكين تحقق الهوية الأساسي وعدم الاعتماد على مصدر الاتصال فقط كمعرف الهاتف مثلاً في حال الاتصال عن بعد.

يتلخص السيناريوالثاني الذي يمكن الحديث عنه في WMLScript وCrypto API. باستخدام التابع signText ضمن الواجهة البرمجية Crypto يمكن إنشاء توقيع رقمي يفتح الباب للبنية التحتية للمفتاح العام PKI لإدارة وإصدار شهادات مفتاح عام. تسمح هذه التقنية بتشفير نهاية إلى نهاية بين مزود الخدمة والزبون.

القسم السادس عشر:

الخدمات المرتبطة بالموقع LBS

الكلمات المفتاحية:

موقع، إحداثيات، تثليث، محطة، خدمات

ملخص:

سنتعرف في هذه الجلسة على ماهية الخدمات المرتبطة بالموقع وما هي الأجيال التي مرت وتمر بها هذه الخدمة. كما سنتعرف على أهم التقنيات المستخدمة في تحديد الموقع ومدى دقتها. وسنتطرق فيه إلى الحديث عن أنظمة المعلومات الجغرافية.

أهداف تعليمية:

يتعرف الطالب في هذا الفصل على:

- مفهوم الخدمات المرتبطة بالموقع والأجيال التي مر بها.
- أهم التقنيات المستخدمة في تحديد موقع الأجهزة المحمولة.
- أنظمة المعلومات الجغرافية وعلاقتها بالخدمات المرتبطة بالموقع.
- ملاحظات حول تطوير التطبيقات المرتبطة بالموقع.

الخدمات المرتبطة بالموقع (LBS)

تعطي إضافة المعلومات المتعلقة بالموقع بعداً هاماً جديداً للعديد من الحلول. إن التطبيقات التي تأخذ الموقع بعين الاعتبار تزيد بشكل كبيرمن فعالية المستخدم وذلك بتقديم وصول مخصص للبيانات ليس مبنياً فقط تفضيلات المستخدم بل على موقع وجود المستخدم أيضاً بما ينقل تخصيص المحتوى إلى درجة جديدة توفر فوائد لا تحصى للمستهلك ولتطبيقات المؤسسة.

الخدمات المبنية على الموقع : ماذا، لماذا ومتى:

يتفق المحللون، ومزودوالخدمة، والمستهلكون على شيء واحد وهوأن الخدمات المبنية على الموقع هي خدمات ذات فائدة عالية بالنسبة لمستخدمي الأجهزة المحمولة.

تتدرج حاجات الأشخاص لقدرات التطبيقات المرتبطة بالموقع من حاجات الأمان وحتى حاجات التجارة المحمولة.

ماذا؟ باختصار الخدمات المبنية على الموقع هي عبارة عن تطبيقات تستفيد من معلومات الموقع لتؤمن الخدمات المناسبة.

لماذا؟ يمكن أن تتدرج هذه الخدمات من إيجاد أقرب محطة وقود إلى الحصول عن معلومات الانتقال بين نقطتين A وB أومعلومات الموقع لأقرب مركز طوارئ.

قد تقوم أنظمة أخرى مرتبطة بالموقع بإرسال رسائل تنبيهية إلى جهاز محمول حول تخفيضات على الأسعار في مركز ما لدى المرور بقربه.

متى؟ يتدرج السوق في هذه المرحلة من الجيل الثاني إلى الثالث وتقود عملية التحول هذه دواعي مرتبطة بالأمان أوالتجارة.

هناك ثلاث أجيال أساسية للخدمات المرتبطة بالموقع :

- الجيل الأول: كانت هذه التطبيقات تتطلب من المستخدم إدخال المعلومات حول موقعه إلى الجهاز. يمكن أن يكون أحد الأشكال الممكنة للإدخال لهذه المعلومات على شكل عنوان بريدي واعتماداً على هذه المعلومات يقدم التطبيق معلومات مخصصة كاتجاهات القيادة والمطاعم القريبة..إلخ.
- الجيل الثاني: يمكن لهذه التطبيقات أن تحدد معلومات الموقع دون الحاجة إلى تدخل مستخدم الجهاز المحمول. يكون هذا الموقع دقيقاً بفارق مسافة لا تزيد عن عدة كيلومترات. تقدم تطبيقات هذا الجيل خدمات مشابهة لما تقوم به تطبيقات الجيل الأول.
- الجيل الثالث: يمكن لهذه التطبيقات الحصول بشكل أدق على معلومات الموقع وتستطيع تزويد خدمات اعتماداً على تلك المعلومات. يمكن لهذه التطبيقات تقديم تحديثات شبه آنية عن الخدمات المحيطة كمعلومات التوجه والطرق ومعلومات المتابعة. تقدم هذه التطبيقات مزايا عديدة ومفيدة للمستخدم لكن ما يزال البعض قلقاً بخصوص قضاء مثل هذه التطبيقات على الخصوصية.

التطبيقات المرتبطة بالموقع

سنستعرض أهم التطبيقات التي تستثمر فيها هذه الخدمات وماهي القيمة التي تعود بها على المستهلكين، الشركات والتطبيقات الحكومية. من المتوقع أن يتوسع تبني هذه التطبيقات مع إدراك المستهلكين والشركات الفائدة التي تمنحها. ولكن قبل أن يحدث هذا لا بد لأ تطبيقات LBS أن تصبح دقيقة، سريعة وسهلة الاستخدام.

في ما يلي أهم وأكثر الاستخدامات شيوعاً لهذه التطبيقات :

خدمات الطوارئ:

في حالات الطوارئ يمكن للمستخدمين الاتصال بمركز طوارئ ليتم تحديد موقعهم وإيصال المساعدة إليهم. وهذه الخدمة هي أساس خدمة E911 المتوفرة في أجزاء من أمريكا الشمالية، كما تستخدم هذه الخدمة بشكل عام لمساعدة العربات المعطلة والتي تواجه مشكلة ما.

معلومات السير المرورية:

تقوم بعض تطبيقات LBS بالمساعدة على تحديد الموقع الحالي للمستخدم وموقع وجود اختناق وتقوم على أساس هذه المعلومات باقتراح المسار الأفضل لتجاوز الاختناق المروري.

التجوال:

يندرج تحت هذا التصنيف العديد من التطبيقات ولكن كلها تتركز على الإجابة عن السؤال : كيف أستطيع أن أصل من النقطة "أ" إلى النقطة "ب"؟ أوعن سؤال أكثر تعقيداً: ما هوالمسار الأمثلي الواجب اتباعه أثناء عملية توزيع منتج على عدة مراكز؟

إدارة الخدمات الميدانية:

تجعل هذه الخدمة من الممكن تتبع منتدبي العملاء في الخدمات الميدانية بحيث يتم تكليف أقرب عميل إلى طلب الخدمة الأقرب إلى موقع وجوده. تفيد هذه الخدمة بشكل كبير العاملين في مجال الخدمات الطارئة كعربات الشرطة والإسعاف والإطفاء.

إدارة الأساطيل:

باستخدامLBS يصبح من الممكن تتبع موقع أساطيل تسليم المنتجات مثلاً وتقدير وقت الوصول التقريبي.

تتبع الممتلكات:

هذه الخدمة شبيهة بالخدمة السابقة ولكنها موجهة لتتبع الممتلكات كالطرود والتجهيزات المحمولة أوالأمتعة ذات القيمة العالية.

الإعلانات اللاسلكية:

يمكن للمستخدمين التسجيل في خدمات تعلمهم بمعلومات متعلقة بمنتج معين خلال وجودهم في منطقة معينة. فعلى سبيل المثال عند الاقتراب من مركز تسوق يمكن أن يتم إعلام المستخدم أن المحل المفضل لديه يعرض تنزيلات. تتطلب هذه الخدمة عادة معلومات دقيقة جداً عن الموقع لذلك تحتاج إلى فترة لتنتشر بصورة فعلية.

خدمات البحث:

تمكن هذه الخدمة مستخدم الجهاز المحمول من إيجاد خدمة أومكان ما في المنطقة التي يتواجد فيها كالعثور مثلاً على مطاعم أومحطات وقود أومساحات لعب غولف. كما يمكن ربط مثل هذه الخدمات بتطبيقات خاصة بالحجز والشراء لتسهيل التجارة المحمولة.

متابعة موقع العربات أوتوماتيكياً:

أحد الخدمات الشائعة التي تم تطبيقها مؤخراً هي خدمة متابعة العربات حيث يمكن بهذه الطريقة تحديد موقع العربات المسروقة واستعادتها.

الخرائط :

يتوفر ضمن العديد من العربات مؤخراً خدمات خرائط تمكن من الحصول على معلومات حول الموقع الحالي وخرائط تفصيلية حول المرافق المحيطة ومؤخراً تم دمج هذه الخدمات مع خدمات التجوال.

معلومات الطقس:

يمكن أن يتم تزويد معلومات عن الطقس بحسب موقع المستخدم ولكن هذه المعلومات ستتعلق دقة المعلومات بدقة أقرب محطة رصد جوي.

تقنيات تحديد موقع الأجهزة المحمولة

قبل إنشاء الحلول المرتبطة بموقع الأجهزة المحمولة لا بد من أن نجد طريقة لتحديد موقع هذه الأجهزة. تتوفر العديد من التقنيات التي تم اعتمادها والتي توفر هذه المعلومات.

إن قرار اختيار أحد هذه التقنيات ليتم استخدامها هوقرار يبنى عادة على موازنة بين عاملين هما الكلفة والدقة. فمع ارتفاع الدقة المطلوب الحصول عليها ترتفع الكلفة المتوقعة للحل.

عادة ما تتوزع هذه الكلفة بين مستخدمي الأجهزة المحمولة ومزود الخدمة اللاسلكية. يجب على المطورين عادة الاعتماد على المعلومات التي تزويدهم بها من الجهاز المحمول ومن مزود الخدمة اللاسلكية وضعف هذه المعلومات سيؤثر بصورة مباشرة على دقة تحديد معلومات الموقع.
في معظم الحالات تتعلق دقة تحديد الموقع بنوع تقنية تحديد الموقع المتبعة.

هناك حلول مبنية على الشبكة يمكن تطبيقها من قبل مزود الخدمة اللاسلكية لتحديد معلومات الموقع للتجهيزات المحمولة القديمة والحديثة.تعتبر هذه الحلول مجدية من حيث الكلفة ولكن الدقة التي تقدمها منخفضة تتراوح بين مئات الأمتار وحتى بضع كيلومترات وذلك بحسب الحل.
أما الحلول المبنية على التجهيزات المخحمولة نفسها فتستطيع تحسين الدقة بشكل كبير ولكنها بالمقابل تؤثر بشكل كبير في رفع الكلفة سواء بالنسبة للجهاز المحمول نفسه أوبالنسبة لمشغل الشبكة.باستخدام تلك الحلول يمكن الحصول على معلومات الموقع بدقة تصل حتى بضعة أمتار أوحتى أقدام من موقع وجود المستخدم.

في معظم الحالات يشكل الحل المختلط التقريب الأفضل وذلك للحصول على حل بدقة معقولة وبكلفة مقبولة.

يعتبر الهدف الأساسي لجميع تقنيات تحديد الموقع تحديد الإحداثيات x,y للجهاز المحمول وسنحاول خلال الشرائح التالي استعراض الطرق الأساسية لإتمام هذا العمل.

تقنيات تحديد موقع الأجهزة المحمولة

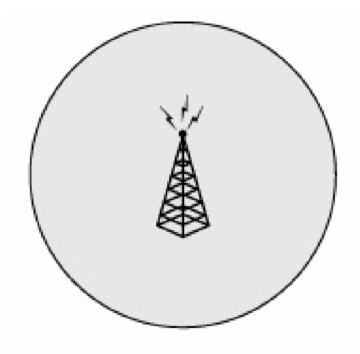
الحلول المعتمدة على الشبكة

يُعتبر استخدام المحطات الثابتة التي تؤلف حامل الشبكة اللاسلكية إحدى الطرق لتحديد موقع زبون محمول. يحتوي كل من هذه المحطات على جهاز اعتراض راديوي لاستقبال الإشارات من الأجهزة المحمولة.
بأخذ الإشارة من محطة واحدة أوأكثر يمكن تحديد موقع الجهاز المحمول. بصورة عامة كلما ازداد عدد المحطات المستخدمة كلما تم الحصول على معلومات أدق حول الموقع.
يمكن للحل المتعمد على الشبكة العمل مع التجهيزات المحمولة الموجودة في الاستخدام وجعلها خطوة أولى لتزويد معلومات الموقع.

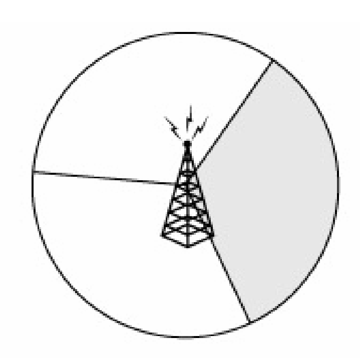
مُعرِّف الخلية:
تعتبر هذه الطريقة من أبسط الطرق لتزويد معلومات الوقع وأكثرها جدوى من حيث الكلفة. وهي ببساطة تحدد أي الخلايا اللاسلكية يستخدمها التجهيز المحمول وتقوم بالإبلاغ عن موقعها. ولما كانت المحطة الأساسية لكل خلية ثابتة فإنه يمكن تحويل معرف الخلية إلى معلومات موقع للجهاز المحمول.
يكمن الجانب السيء لهذا الحل في أن الموقع الدقيق للجهاز المحمول ضمن مجال الخلية غير محدد.
توفر هذه الطريقة معلومات بفارق حتى 1-2 كيلومتر. تعتبر هذه المعلومات مقبولة لأخذ فكرة عامة عن الموقع ولكنها لا توفر تلك المعلومات الكافية لخدمات الطوارئ أوالتوجيه على الطرق أوالإعلانات.
لحسن الحظ هناك طرق أخرى لتحسين الدقة التي توفرها طريقة محدد الخلية.
بعض الخلايا مقسم إلى قطاعات ويقلل المساحة الكلية وهذا ما يمكن أن يقلل الخطأ في تحديد الموقع حتى الثلث.

للحصول على المزيد من الدقة يمكن استخدام تقنية تسمى TA والسبق الزمني والتي تساعد في معرفة بعد المستخدم عن المحطة وبالتالي تقليل الخطأ في تحديد الموقع بشكل كبير.
لا تعتبر هذه الطريقة دقيقة ولكنها تحسن الدقة العامة في تحديد الموقع باستخدام معرف الخلية.
للأسف فإن معلومات الحصول على معلومات TA ليس بالأمر السهل دون الوصول إلى مركز تحديد المواقع المحمولة MPC والذي يمكن أن يوفر معلومات مفصلة عن الموقع باستخدام واجهات تطبيقات برمجية API. يمكن للمطورين كتابة تطبيقات للاتصال بـMPC للحصول على معلومات TA مع معلومات محدد الخلية.

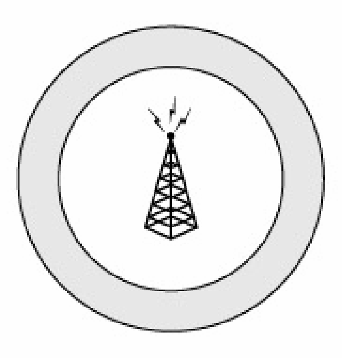
يوضح الشكل التالي الآليات المستخدمة التي ناقشناها ضمن طريقة مُعرِّف الخلية:



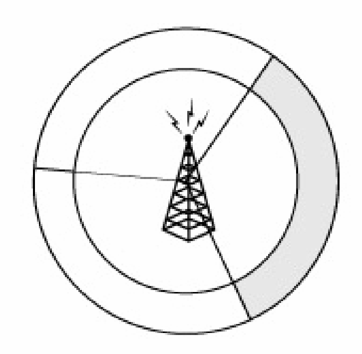
Cell Identity
with an omnisector cell



Cell Identity
with a three-sector cell



Cell Identity
with timing advance



Cell Identity
with a three-sector cell
and timing advance

تسمى هذه الطرق عند تركيبها معاً CGI-TA وهي اختصار **لمعرف الخلية العام–السبق الزمني** يمكن لهذا التقريب أن يقودنا إلى نتائج بدقة بين 100 إلى 200 متر. وهي عبارة عن دقة جيدة جداً نسبة إلى بساطة هذه التقنية التي لا تتطلب أي تحديث في الأجهزة المحمولة.

لا بد من الذكر أيضاً بأن هذه الطريقة أكثر فعالية ضمن المدن نظراً لكثافة المحطات في المناطق المأهولة.

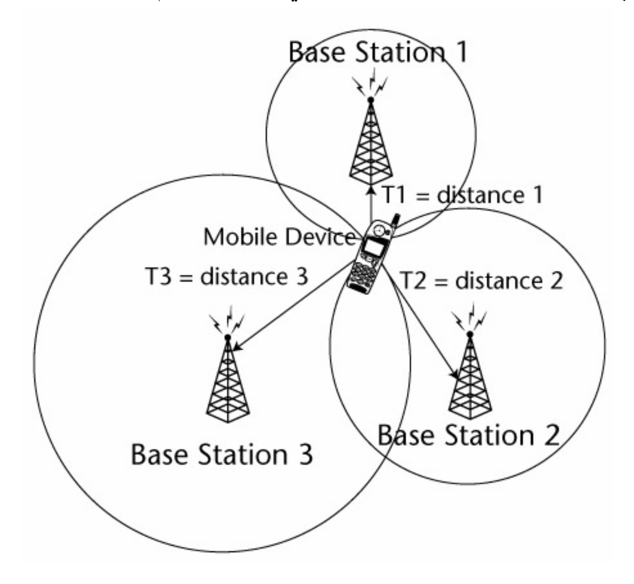
تقنية زمن الوصول TOA

حتى عند اعتماد تقنية السبق الزمني مع معرف الخلية لا نحصل على الدقة الكافية للعديد من التطبيقات والخدمات التي قمنا باستعراضها في بداية الجلسة.

التقريب الآخر في هذا المجال يدعى تقنية زمن الوصول TOA أوفرق زمن الوصول TDOA.

يمكن لهذه التقنية أن تحسن بشكل كبير دقة تحديد الموقع. بدلاً من استخدام محطة واحدة لتحديد موضع الجهاز المحمول تستخدم TAO معلومات تم جمعها من ثلاث محطات أوأكثر. تعمل هذه الطريقة بجعل الجهاز المحمول يرسل إشارة يتم استقبالها من جميع المحطات ضمن مدى الجهاز. تقوم كل محطة بعدها بحساب الزمن الفاصل بين إرسال الرسالة واستقبالها وليكن لثلاث محطات مثلاً (T1,T2,T3) يجب أن تكون هذه الأزمان دقيقة جداً تتطلب كون الوقت في جميع المحطات يخضع لعلمية مزامنة. وهذا ما يتطلب إما نظام GPS أوساعة ذرية وكلا الحلين يعتبر مكلفاً كما نعلم.

بما أن الإشارة تتحرك بسرعة ثابتة فإنه سيصبح بالإمكان تحديد مسافة الجهاز المحمول عن المحطة. ولكن المسافة المحددة من خلية واحدة ليست ذات فائدة ولكن باستخدام المعلومات من ثلاث محطات يمكن تحديد مكان الجهاز المحمول نسبة إلى مواقع المحطات وبعدها يمكن ترجمة الإحداثيات النسبية للجهاز نسبة إلى المحطات إلى إحداثيات عامة تحدد الموقع.
لا بد هنا من التنويه إلى أن استخدام تقنية TOA عملية جداً في الشبكات التي تعتمد تقنية CDMA/CDMA2000 لأن هذه الشبكات مزامنة منذ البداية ولا تحتاج إلى أي ساعة ذرية أوGPS. الشكل التالي يبين استخدام تقنية زمن الوصول لتحديد الموقع



تقنية زاوية الوصول AOA:

تعمل هذه التقنية بشكل مشابه لتقنية TOA ولكن عوضاً عن استخدام الزمن اللازم لإشارة يطلقها الجهاز المحمول للوصول إلى ثلاث محطات تستخدم هذه التقنية الزاوية أتت منها الإشارة إلى المحطة. بمقابلة هذه المعلومات من ثلاث محطات على الأقل يمكن تحديد موضع الجهاز المحمول.
تستخدم بعض الأنظمة تقنية زاوية الوصول مع تقنية زمن الوصول للحصول على دقة أعلى.

الحلول المبنية على الجهاز المحمول نفسه

عندما يكون المطلوب الحصول على دقة أعلى في تحديد الموقع لا بد من استخدام الحلول المبنية على الجهاز المحمول نفسه.
في هذه الحلول يشترك الجهاز المحمول بصورة أساسية في عملية تحديد الموقع. تسمح دقة هذه الحلول بتقديم خدمات الجيل الثالث من LBS والتي تتطلب تحديد موقع دقيق.
يعتمد الحلان الذين سنقوم بعرضهما على نفس الطريقة لحساب الموقع ولكن الاختلاف الأساسي بينهما يكمن بأن E-OTD تعتمد على المحطات الأرضية في حين تعتمد GPS على الأقمار الصناعية.

التقنية المحسنة للاختلاف الزمني (E-OTD):

إن تقنية E-OTD تعمل بشكل مشابه لمبدأ عمل TOA ولكن يتولى الجهاز المحمول حساب الزمن بدلاً عن المحطة. تعتمد هذه التقنية على حساب الزمن اللازم للإشارة القادمة من محطة في مكانين جغرافيين مختلفين. يشكل الجهاز المحمول المحطة الثابتة التي يعمل معها ما يسمى ب LMU أووحدة قياس الموقع.

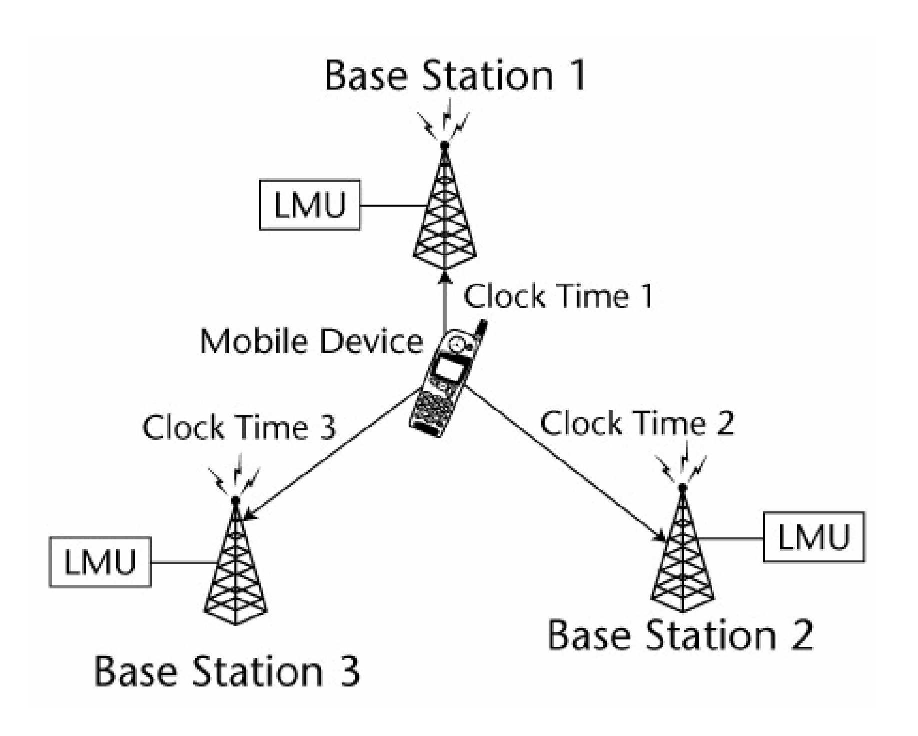
للحصول على موقع دقيق يجب أن تشترك المعلومات من ثلاث محطات في عملية الحساب تلك. لتنجح هذه التقنية يجب على المحطات إرسال التوقيت بشكل دقيق إلى الجهاز المحمول. باستخدام هذا التقريب، يجب إرسال جميع الإشارت في نفس الوقت لأن مستخدم الجهاز المحمول قد يكون في حالة حركة أثناء القياس وهنا يبرز دور LMU حيث تزود الأخيرة مصدر توقيت دقيق من أجل عملية القياس للتأكد من دقة المعلومات.

ما أن تتم عملية أخذ القياسات حتى يقوم الجهاز المحمول الذي يدعم تقنية E-OTD بتسجيل فرق الزمن من المحطات الثلاثة. عندها يمكن حساب المسافة بين الجهاز المحمول والمحطات وذلك بمقارنة فرق الزمن بين قياسات الزمن.يمكن تحويل فرق الزمن إلى مسافة لأن الإشارة تتحرك بسرعة ثابتة. بعد الحصول على المسافات يمكن الحصول على الموقع النسبي حيث يمكن لهذه العملية أن تتم على الجهاز المحمول نفسه أومن قبل المحطة وبعدها يمكن حساب الإحداثيات المطلقة اعتماداً على موقع المحطات.

يقوم الجهاز المحمول بالقيام بهذه الحسابات اعتماداً على حل برمجي ولكن لا بد من عملية تحديث البنية الصلبة للجهاز أيضاً لدعم E-OTD.

مما سبق نرى أن تقنية E-OTD توفر حل دقيق وبكلفة مقبولة. حيث يمكن الوصول إلى دقة تصل حتى 50 إلى 100 متر.

يوضح الشكل التالي بنيان تقنية E-OTD:



GPS وA-GPS

يُعتبر نظام تحديد الموقع العالمي GPS من أكثر التقنيات شعبية في الوقت الحاضر. يستخدم هذا النظام 24 قمر صناعي عالمي تشكل مدار حول الأرض لإرسال إشارات إلى المستقبلات التي تدعم GPS.

يمكن للمستقبل الاتصال مع ثلاث أوأربع أقمار صنعية في أي لحظة.

لتصبح هذه العملية ممكنة يجب أن يكون هناك خط نظر بين المستقبل والقمر الصنعي وهذا ما يمنع استخدام GPS ضمن الأبنية.

عند حصول المستقبل على قياسات الموقع يمكنه حساب الإحداثيات مباشرة أوإرسال المعلومات إلى مخدم الشبكة لمعالجتها.

بشكل مشابه لما يحصل في تقنية E-OTD تعتبر عملية الحساب معقدة نسبياً وتحتاج إلى قدرة معالجة لا بأس بها.وإذا كانت الحسابات ستتم على الجهاز المحمول نفسه يجب على المصنع تزويد الجهاز بميزات في الكيان الصلب لتوفر قدرة المعالجة المناسبة مما يحمل المزيد من الكلفة في سعر الجهاز المحمول.

في العديد من التجهيزات المحمولة يقدم مستقبل GPS كوحدة منفصلة قابلة للوصل مع الجهاز المحمول باستخدام كابل أوبصورة لاسلكية باستخدام تقنية بلوتوث.
بمعنى أن وحدة GPS تحتوي الكيان الصلب المطلوب دون التأثير بصورة مباشرة على شكل الجهاز ة على استهلاكه للطاقة.

إن أسعار الشرائح الصغرية الخاصة ب GPS واستهلاكها للطاقة قد انخفض بشكل كبير في الآونة الأخيرة بشكل يقود إلى استخدام هذه التقنية بصورة مبيتة ضمن التجهيزات المحمولة.
يعمل GPS بشكل مشابه لتقنيات تحديد الموقع المعتمدة على التثليث , حيث تقوم الأقمار الصنعية ببث إشارات يمكن قراءتها من قبل التجهيزات التي تدعم GPS.
ليس من المهم للقمر الصنعي عدد الأجهزة التي تستقبل هذه الإشارة لأن الاتصال يتم باتجاه واحد فقط.

يقوم الجهاز المحمول بقياس كمية الزمن التي تتطلبها إشارة القمر الصنعي للوصول إلى الجهاز. تؤخذ هذه القياسات من ثلاثة أقمار مختلفة لإعطاء معلومات موقع أكثر دقة.
رياضياً تتطلب العملية معلومات من أربعة أقمار لكن القياسات من ثلاثة أقمار كافية للحصول على معلومات لإعطاء دقة كافية.ولما كانت سرعة الإشارة محددة يمكن عندئذ تحديد المسافة من القمر الصنعي.
كما نلاحظ فمن الضروري أن تكون قياسات الزمن دقيقة جداً فالخطأ في جزء واحد من ألف من الثانية قد يقود إلى خطأ في الموقع يصل إلى 300 كيلومتر. لهذا السبب تستخدم مستقبلات GPS المعلومات القادمة من ساعة ذرية في كل قمر صنعي للتأكد من أن الزمن دقيق.
بعد حساب المسافة عن القمر الصنعي يمكن أن تتم عملية التثليث وتحديد الإحداثيات المطلقة للجهاز المحمول. تزود هذه التقنية نتائج بدقة تتراوح بين 5 وحتى 40 متراً عن الموقع الأصلي.
يجب ألا ننسى أيضاً أن معلومات الموقع التي تزودها أنظمة GPS تتضمن إحداثيات الطول والعرض والارتفاع أيضاً.

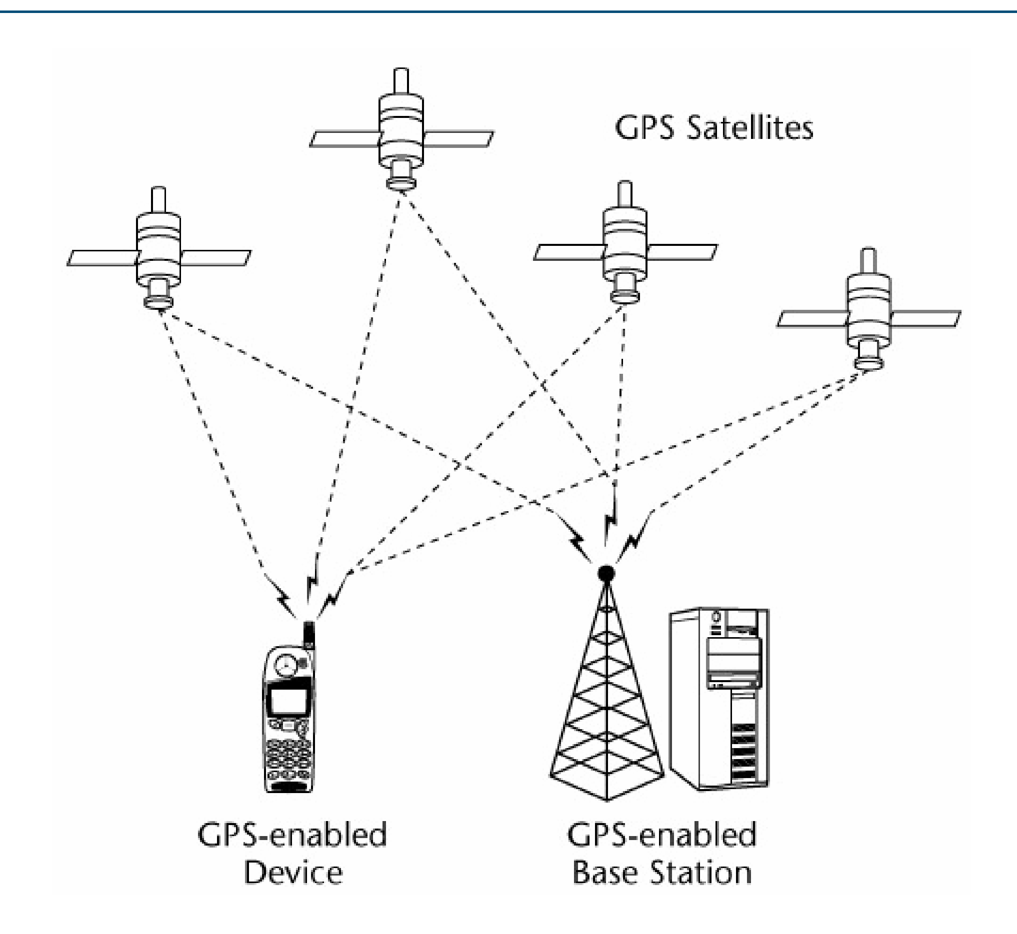
بالرغم من كون الأنظمة التي تعتمد GPS تزود معلومات موقع دقيقة لكنها ليست متفوقة بشكل كبير. لأنه كما ذكرنا سابقاً وللحصول على قراءة يجب أن يكون هناك خط نظر مع القمر الصنعي. وهذه النقطة تعد محدودية كبيرة بالنسبة للأعمال المحمولة.فالعديد من التطبيقات تتطلب الاستخدام ضمن الأبنية أوالعربات مما يجعل من الصعب على GPS تزويد الخدمة المطلوبة.قد تكون أحد الحلول لنقطة الضعف هذه استخدام طريقة TOA–CGI كطريقة داعمة.

أحد المشاكل الأساسية الأخرى هي الزمن اللازم لتحديد الموقع باستخدام هذه التقنية قد يتراوح بين 20 إلى 40 ثانية وهورقم كبير جداً يحول دون استخدام العديد من التطبيقات.
أحد الحلول التي تتجاوز نقطتي الضعف الأساسيتين في GPS هي استخدام ما يسمى A-GPS أونظام تحديد الموقع العالمي المدعوم حيث يستخدم هذا النظام أجهزة محمولة معدلة تقوم باستقبال معلومات GPS ثم إرسال القراءات إلى مخدم الشبكة. يقوم الأخير باستخدام مستقبلات GPS أخرى (وهي تشكل جزء من الشبكة) لإعانة المعلومات التي يقدمها GPS الخاص بالجهاز المحمول.

يتم توزيع مستقبلات GPS الخاصة بالشبكة عبر الشبكة بمسافات تقدر بمئات الكيلومترات. تقوم هذه المستقبلات باستقبال معلومات GPS المرسلة من قبل القمر الصناعي وتقوم بتزويد الأجهزة المحمولة بهذه المعلومات بشكل يزودهم بالقدرة على حساب الفوارق الزمنية دون الحاجة إلى فك تشفير الرسالة القادمة مباشرة من القمر الصنعي.
تخلق هذه العملية فرق كبير جداً في الزمن اللازم للحصول على معلومات الموقع حيث يصل الزمن المطلوب إلى حوالي ثماني ثواني.
أما بشأن المحدودية المتعلقة بخط النظر فيقوم الجهاز المحمول بإرسال معلومات القياسات إلى مخدم الشبكة بحيث تتم عمليات الحساب المعقدة عليه. تسمح قدرة المعالجة بحسابات تقليل المسارات المتعددة وتحليل الإشارة لتحديد موقع الجهاز ضمن المبنى وفي الأماكن التي يصعب فيها تحديد الموقع بالأسلوب التقليدي.

تستخدم تقنية مستقبلات GPS بصورة مرتبطة مع أنظمة المعلومات الجغرافية لتزويد مشهد طبوغرافي للمناطق المختلفة ضمن البلاد.
أحد أكثر استخدامات تطبيقات GPS شيوعاً هي نظام تحديد الموقع الخاص بالعربات وخدمات التتبع. تتطلب هذه الخدمات أكثر من مجرد تحديد الموقع لجعلها فعالة وهنا يأتي دور أنظمة المعلومات الجغرافية.

يوضح الشكل التالي بنيان تقنية A-GPS



ما هو نظام المعلومات الجغرافية GIS

يشكل تحديد موقع الأجهزة المحمولة أحد الجوانب فقط في عملية تزويد الخدمات المرتبطة بالموقع. فالحصول على إحداثيات الموقع ضروري بالطبع ولكن ما نفعله بتلك الإحداثيات لا يقل أهمية.

نظام المعلومات الجغرافية هوعبارة عن برنامج خاص بوضع الخرائط بإمكانه ربط معلومات الموقع بمعلومات أخرى ذات صلة ليعطي معلومات الموقع قيمة ومعنى.

يتم الوصول إلى هذا عادة باستخدام خرائط متعددة الأبعاد للمعلومات تتضمن مثلاً لمواقع الأبنية، ولتوضع الشوارع، ولكثافة السكان وللكثير من المعلومات الأخرى.

يتألف نظام المعلومات الجغرافية المتكامل من الكيانات الصلبة، والبرمجيات، والبيانات، وأشخاص مدربين يعلمون كيفية إجراء عمليات التحليل على معلومات التي يزودها النظام.

تفيد أنظمة GIS في تحقيق غرضين أساسيين:

- إيجاد معالم معينة: تفيد هذه العملية في إيجاد معلومات عن مكان معلّم ما أوعن ماهيته. تتضمن هذه العملية إيجاد أقرب مطعم

أومحطة وقود مثلاً أوتحديد المسار الأفضل للوصول إلى مكان محدد.

- إيجاد نموذج أوتشكيل ما: ترتبط هذه القدرة بتحليل الأعمال وهي لا تُستخدم فقط من التجهيزات المحمولة. تهتم الأعمال عادة بمعرفة توزيع المعلومات بدلاً عن المعلومات الإفرادية عن معلم ما. على سبيل المثال عند محاولة تحديد مكان لعقد مؤتمر مثلاً قد تكون أحد المعلومات الهامة تحديد المنطقة التي يتواجد فيها عالبية الأشخاص المستهدفين بهذا المؤتمر.

من الواضح تماماً أن فائدة أنظمة GIS كبيرة للتطبيقات المحمولة ولكن هذه الأنظمة تقدم فوائد تتجاوز ما هومطلوب للبيئة المحمولة فيما يلي أهم استخدامات هذه الأنظمة:

- إيجاد ما يوجد في الجوار: يعد هذا الاستخدام من أكثر الاستخدامات شيوعاً ولمستخدمي التجهيزات المحمولة. بإعطاء موقع محدد يقوم النظام بإيجاد كل المراكز ضمن نصف قطر محدد. قد تتضمن هذه المراكز المراكز الصحية، المطاعم، محطات الوقود، أوحتى مراكز البيع لمواد معينة.
- معلومات التوجه: وهي أحد الاستخدامات المهمة أيضاً بالنسبة لمستخدمي التقنيات المحمولة.
- المعلومات التنبيهية: حيث قد يرغب بعض المستخدمين بالحصول على معلومات تهمهم حين تصبح قريبة من موقع تواجدهم على سبيل المثال تنبيه بوجود ازدحام مروري على الطريق الذي يعبره المستخدم.
- كثافة التموضع: وهي معلومات تحليلية قد تكون مفيدة للكثير من الأعمال كتحديد مواقع الكثافة الأعلى للجرائم على سبيل المثال.
- كميات التموضع: وذلك لتحديد أين تنحصر مناطق التوزع الأعظمية والأصغرية لخدمة ما.

تطوير أنظمة الخدمات المتعلقة بالموقع

فتحت الخدمات المتعلقة بالموقع الباب أمام أسواق جديدة لكن هذه الخدمات تتطلب بيئة عمل معقدة تستخدم تقنيات متعددة وطرق مختلفة قد تقود إلى نفس النتيجة.

ومع عدم توفر خبرات متراكمة في هذا الموضوع بعد يصبح موضوع تطوير تطبيقات LBS موضوعاً صعباً نسبياً.

يقدم مزودوا الخدمة حالياً واجهات برمجية للتطبيقات API للوصول إلى معلومات الموقع وهي بصورة عامة واحدة من نوعين:

- واجهات التطبيقات البرمجية المبنية على الشبكة: حيث تتوفر معلومات الموقع على مخدم الشبكة ويمكن الوصول إليها من أي طرف يمتلك الصلاحية المناسبة.
- واجهات التطبيقات البرمجية المبنية على الجهاز المحمول: حيث يتم توليد معلومات الموقع على الجهاز ويمكن الوصول إليها بشكل مباشر عن طريق التطبيقات العاملة على الجهاز.

توفر أغلب الحلول المتعلقة بالموقع واجهات برمجية للمطورين. يتم الوصول إلى معلومات الموقع عن طريق مركز تحديد الموقع المحمول MPC. يتم استخدام نفس الواجهة البرمجية بغض النظر عن التقنية المستخدمة في تحديد الموقع وبالتالي توفير طبقة تجريدية للمطورين.

يعد وجود MPC خطوة بالاتجاه الصحيح بالنسبة للمطورين ولكنها لا تحل كل المشاكل. تظل كثير من التحديات التي يجب تجاوزها وبالأخص الحاجة إلى التقييس والتجوال بين الخدمات التي يقدمها المشغلون المختلفون.

القسم السابع عشر والثامن عشر

تمارين وأمثلة

الكلمات المفتاحية:

نص برمجي، تطبيق.

ملخص:

سنتعامل في هذه الجلسة العمل مع بعض التمارين المتكاملة التي تهدف إلى صقل الخبرات التطبيقية لبعض الأفكار التي تمت تغطيتها خلال جلسات هذه المادة.

أهداف تعليمية:

يتعرف الطالب في هذا الفصل على:

- كيفية استخدام Visual studio للوصول إلى تطبيق عامل
- بعض التقنيات التطبيقية الخاصة باستخدام ملفات Xml وقواعد بيانات MSSQL و MS Access كمصادر للبيانات.

أمثلة عامة

تم تخصيص الجلستين الأخيرتين في هذه المادة لاستعراض مجموعة من الأمثلة المتكاملة التي تستخدم التقنيات التي تعرفنا عليها ضمن الجلسات الماضية ضمن إطار حل بسيط متكامل.

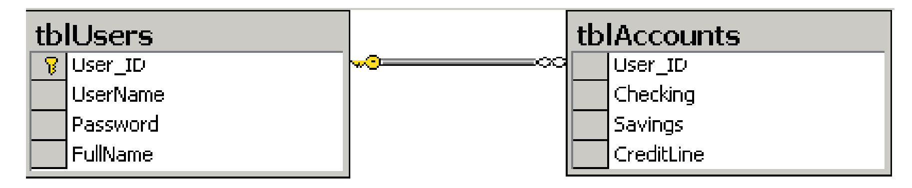
المثال الأول :

المتطلبات اللازمة للعمل على هذا المثال هي:

- توفر بيئة التطوير Visual Studio .NET
- Microsoft Mobile Internet Toolkit والتي يمكن الحصول عليها بتحميلها عن طريق الرابط http://www.microsoft.com/downloads/details.aspx?familyid=ae597f21-b8e4-416ea28fb124f41f9768&displaylang=en.
- وجود مخدم MS SQL عامل على الجهاز.
- وجود مخدم IIS عامل على الجهاز.

هذا المثال عبارة عن مثال لتطبيق بسيط يحاكي آلة ATM ولكن بدون القدرة على إجراء مناقلات حقيقية. الغرض من هذا المثال إظهار المقدار الحالي للمبالغ المتوفرة في حساب ما.

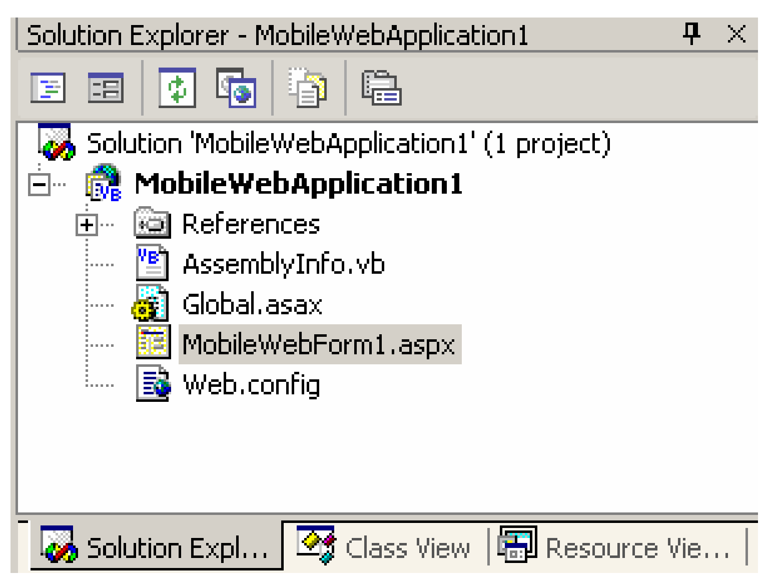
1- قاعدة البيانات: باعتبار أن هدفنا في هذا المثال ليس التركيز على قاعدة البيانات وتصميمها سنقوم باستخدام أبسط شكل يخدم هدف مثالنا:



نلاحظ أننا استخدمنا هنا جدولين الأول خاص بالمستخدمين و الثاني خاص بالحسابات

2- من بيئة VS.NET نقوم باختيار إنشاء مشروع جديد من النمط Mobile Web Application.

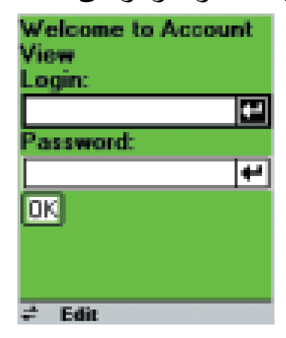
3- سيولد VS.NET تلقائياً مجموعة من الملفات و النصوص البرمجية أهمها الملف MobileWebForm.aspx و MobileWebForm.vb (إذا اخترنا لغة Visual basic للتطوير).



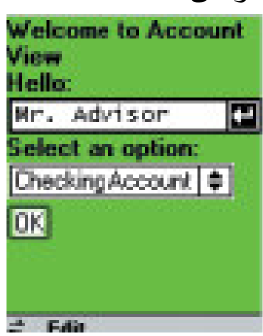
4- باستخدام التبويب الحاوي على عناصر تحكم نماذج الوب المحمولة سنقوم بجر عناصر TextBox و Label و Button .

5- ستتألف صفحة الوب لدينا من ثلاث نماذج

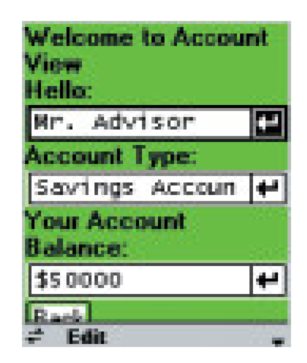
- الأول خاص بإدخال معلومات تسجيل الدخول وهو من الشكل



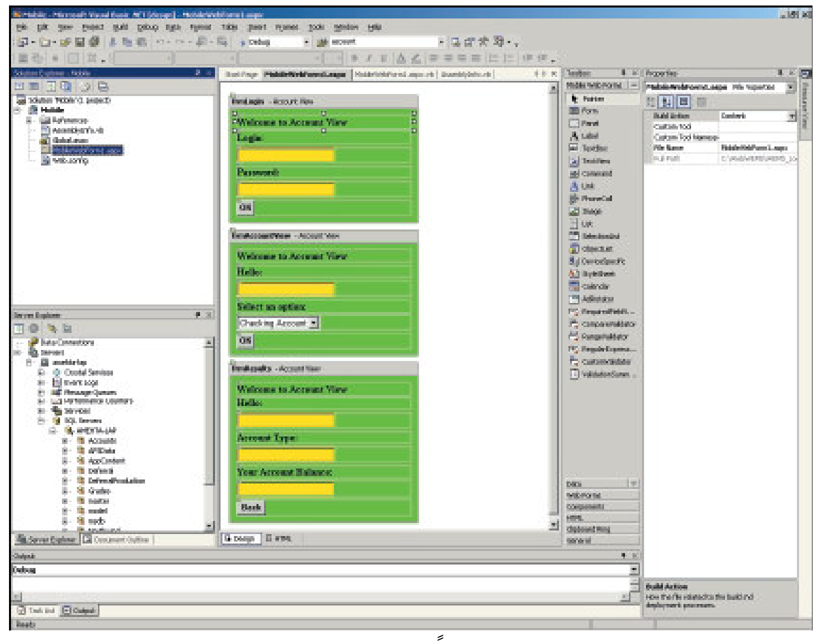
- الثاني خاص باختيار نوع الحساب وهو من الشكل:



- أما الثالث فخاص بإظهار النتائج وهو من الشكل:



سيؤول الشكل العام لواجهة التطوير إلى ما يلي



6- لدى إضافة عناصر التحكم المختلفة يتم تلقائياً إضافة النص البرمجي لصفحة MobileWebForm1.aspx ويصبح النص البرمجي كما يلي:

```
<%@ Page Language="vb" AutoEventWireup="false" Codebehind="MobileWebForm1.aspx.vb"
Inherits="MobileWebApplication1.MobileWebForm1" %>
<%@ Register TagPrefix="mobile" Namespace="System.Web.UI.MobileControls"
Assembly="System.Web.Mobile" %>
<HEAD>
        <meta content="Microsoft Visual Studio .NET 7.1" name="GENERATOR">
        <meta content="Visual Basic .NET 7.1" name="CODE_LANGUAGE">
        <meta content="http://schemas.microsoft.com/Mobile/Page"
name="vs_targetSchema">
</HEAD>
<body Xmlns:mobile="http://schemas.microsoft.com/Mobile/WebForm">
        <mobile:form id="Form1" runat="server">
                <P>Welcome to Account View</P>
                <P>Login
<mobile:TextBox id="txtUserName" runat="server"></mobile:TextBox>Password
<mobile:TextBox id="txtPassword" runat="server"></mobile:TextBox>
<mobile:Command id="cmdOK" runat="server">OK</mobile:Command></P>
        </mobile:form>
        <mobile:form id="frmAccountView" runat="server">
                <P>Welcome to Account View </P>
                <P>Hello
```

```
<mobile:TextBox id="txtAccountViewName" runat="server"></mobile:TextBox>Select An
option
<mobile:SelectionList id="SelectionList1" runat="server">
        <Item Value="0" Text="Checking Account"></Item>
        <Item Value="1" Text="Saving Account"></Item>
        <Item Value="2" Text="Credit Account"></Item>
                        </mobile:SelectionList>
<mobile:Command id="Command1" runat="server">OK</mobile:Command></P>
        </mobile:form>
        <mobile:form id="frmResults" runat="server">
                <P>Welcome to Account View </P>
                <P>Hello
<mobile:TextBox id="txtOutputName" runat="server"></mobile:TextBox>Account Type
<mobile:TextBox id="txtOutputAccountType" runat="server"></mobile:TextBox>Account
Balance
<mobile:TextBox id="txtOutputAccountBalance" runat="server"></mobile:TextBox>
<mobile:Command id="Command2" runat="server">Back</mobile:Command></P>
        </mobile:form>
</body>
```

أما النص البرمجي في الخلفية MobileWebForm1.aspx.vb:

```
'System and SQL Namespaces
Imports System.Data
Imports System.Data.SqlClient
Public Class MobileWebForm1
    'Mobile Namespaces
    Inherits System.Web.UI.MobileControls.MobilePage
    'The source code contains Mobile Control's events in this
    'section such as the two protected WithEvents shown below:

    Protected WithEvents Password As _
    System.Web.UI.MobileControls.Label
    Protected WithEvents Form1 As System.Web.UI.MobileControls.Form
    Protected WithEvents txtUserName As System.Web.UI.MobileControls.TextBox
    Protected WithEvents txtPassword As System.Web.UI.MobileControls.TextBox
    Protected WithEvents SelectionList1 As System.Web.UI.MobileControls.SelectionList
    Protected WithEvents cmdOK As System.Web.UI.MobileControls.Command
    Protected WithEvents frmResults As System.Web.UI.MobileControls.Form
    Protected WithEvents frmAccountView As System.Web.UI.MobileControls.Form
    Protected WithEvents Command2 As System.Web.UI.MobileControls.Command
    Protected WithEvents txtAccountViewName As System.Web.UI.MobileControls.TextBox
    Protected WithEvents Command1 As System.Web.UI.MobileControls.Command
    Protected WithEvents txtOutputName As System.Web.UI.MobileControls.TextBox
    Protected WithEvents txtOutputAccountType As System.Web.UI.MobileControls.TextBox
    Protected WithEvents txtOutputAccountBalance As
System.Web.UI.MobileControls.TextBox
```

قمنا أولاً كما نرى باستيراد فضاء الأسماء والتصريح عن جميع عناصر التحكم المستخدمة.

```
'Following constants define the type of account you want to query
    Public Const CHECKING_Account = 0
    Public Const SAVINGS_Account = 1
    Public Const CREDIT_Line = 2
```

عرفنا هذه الثوابت لغرض استخدامها كبدائل عن استخدام الأرقام في تحديد نوع الحساب.

```
  Private Sub cmdOK_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles cmdOK.Click
        'Get the username,if its a valid username then
        'navigate to account view form
```

```
        Dim strUser As String
        strUser = GetUserName(txtUserName.Text, txtPassword.Text)
        If Len(strUser) > 0 Then
            ActiveForm = (frmAccountView)
        Else
            Response.Write("Invalid Username and/or Password")
        End If
    End Sub
```

الطريقة السابقة خاصة بمعالجة حدث الضغط على زر "OK" في النموذج الأول حيث سيتم اختبار اسم المرور وكلمة السر باستخدام الطريقة GetUserName

```
    Private Function GetUserName(ByVal sUserName As String, ByVal sPassword As
String) As String
        'Get Username based on txtName.Text and
        'txtPassword.Text from tblUsers
        Static strUser As String
        If Len(strUser) = 0 Then
            Dim sql As String = "SELECT * FROM tblUsers Where tblUsers.UserName ='" &
sUserName & "'AND tblUsers.Password =_"
            '"&sPassword &"'"
            'Use ADO.NET Data Reader to get the data

            Dim conn As New SqlConnection("workstation id=SKAIT;packet
size=4096;integrated security=SSPI;data source=SKAIT;persist security
info=False;initial catalog=Accounts")
            Dim comm As New SqlCommand(sql, conn)
            Dim reader As SqlDataReader
            conn.Open()

            reader = comm.ExecuteReader
            While (reader.Read())
                strUser = (reader("FullName"))
            End While
            conn.Close()
            conn = Nothing
            comm = Nothing
            reader = Nothing
        End If
        GetUserName = strUser
    End Function
```

يقوم التابع السابق كما نرى بعملية اتصال مع قاعدة البيانات باستخدام الأغراض sqlConnection، SqlCommand، SqlDataReader نلاحظ هنا أننا استخدمنا SqlDataReader بسبب عدم الحاجة إلى إجراء أي تعديل على البيانات.

```
    Private Sub Command1_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Command1.Click
        'navigate to the results form using Activeform
        ActiveForm = (frmResults)
    End Sub
```

تمثل الطريقة السابقة معالج حدث ضغط زر OK في النموذج الثاني بعد اختيار نوع الحساب.

```
Private Sub frmResults_Activate(ByVal sender As System.Object, ByVal e As
```

```
System.EventArgs)
        'get the data based on account type and username and then
        'format the result
        Dim intOption As Integer
        Dim strOutput As String
        intOption = (SelectionList1.SelectedIndex)
        '0 =Checking,1 =Savings,2 =Credit Line
        txtOutputAccountType.Text() = SelectionList1.Selection.Text()       'Account
Type
        txtOutputName.Text = GetUserName(txtUserName.Text, txtPassword.Text) 'Get the
name
        txtOutputAccountBalance.Text = Format$(Get_Account_Balance(intOption),
"$##.#####,###.00")
        'Get the account value and format the string
    End Sub
```

تمثل الطريقة السابقة معالج حدث تفعيل النموذج frmResults والذي سيقوم بإظهار النتائج.

```
    Private Function Get_Account_Balance(ByVal intAccountType) As Double
        'Classic use of SQL data reader,it is atomic read,
        'closoes 'the connection
        'get the account balance based on relationship with the Table()
        Dim sql As String = "SELECT *FROM tblAccounts,tblUsers Where
tblAccounts.User_ID = tblUsers.User_ID And tblUsers.UserName ='" & txtUserName.Text &
"'AND tblUsers.Password ='" & txtPassword.Text & "'"
        Dim conn As New SqlConnection("Data Source=localhost;Integrated
Security=SSPI;Initial Catalog=Accounts")
        Dim comm As New SqlCommand(sql, conn)
        Dim reader As SqlDataReader
        Dim dblAccountValue As Double
        conn.Open()
        reader = comm.ExecuteReader
        Select Case intAccountType
            Case CHECKING_Account
                While (reader.Read())
                    dblAccountValue = Val(reader("Checking"))
                End While
            Case SAVINGS_Account
                While (reader.Read())
                    dblAccountValue = Val(reader("Savings"))
                End While
            Case CREDIT_Line
                While (reader.Read())
                    dblAccountValue = Val(reader("CreditLine"))
                End While
        End Select
        conn.Close()
        conn = Nothing
        comm = Nothing
        reader = Nothing
        Get_Account_Balance = dblAccountValue
    End Function
```

الطريقة السابقة خاصة بالاتصال بقاعدة البيانات واستخراج المعلومات الخاصة بكل نوع حساب للمستخدم الذي قام بتسجيل الدخول.

```
    Private Sub Command2_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Command2.Click
        'Navigate to account view
        ActiveForm = frmAccountView
    End Sub
```

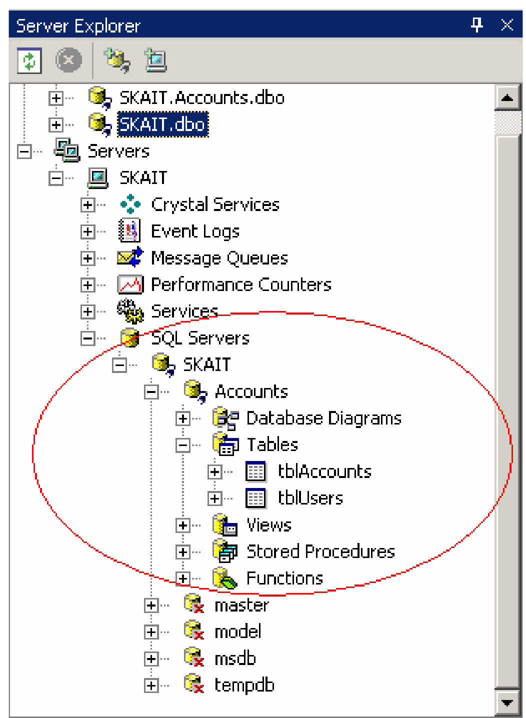
معالج الحدث الخاص بالنقر على زر Back في النموذج frmResults للعودة والاستعلام عن نوع آخر من الحسابات.

```
    Private Sub frmAccountView_Activate(ByVal sender As System.Object, ByVal e As
System.EventArgs)
        'Welcome the user
        txtAccountViewName.Text = GetUserName(txtUserName.Text, txtPassword.Text)
    End Sub

    Private Sub InitializeComponent()

    End Sub
End Class
```

7- لسهولة العمل توفر بيئة VS.NET آلية لإنشاء وتصميم قاعدة البيانات من ضمن بيئة التطوير وذلك من خلال إظهار نافذة Server Explorer:



حيث نلاحظ كيف قمنا هنا بإنشاء قاعدة البيانات Accounts التي تحتوي الجداول المطلوبة

8- يمكن إضافة طرق أخرى لمثالنا تقوم بعمليات السحب من حساب ما مثلاً و ستكون على غرار تلك التي

استعملناها في الاستعلام عن حساب.

9- أصبح بإمكاننا الآن أن نقوم ببناء تطبيقنا باستخدام الأمر Build Solution من القائمة Build.

10- بعد التأكد من كون تطبيقنا موجود ضمن المسار المناسب في مجلد ضمن WWWRoot أو مُعرَّف ضمن مجلد افتراضي من خلال IIS والتأكد من تفعيله كتطبيق يمكننا تشغيل تطبيقنا المصغر عن طريق مستعرضات مختلفة لاختباره.

11- يمكن هنا أيضاً استخدام أنواع مستعرضات غير مدعومة ومحاولة تأمين الدعم لها باستخدام موائمات أجهزة و ضبط الإعدادات من ملف Web.config.

المثال الثاني

سنعمل في هذا المثال على حل يتعلق بتحديث حالة الطقس على الأجهزة المحمولة. سنستخدم في مثالنا ملف XML كمصدر لبيانات حالة الطقس (حيث يمكن أن يتوفر مثل هذا المصدر بسهولة).

للتعامل مع هذا المصدر سنستخدم الصف XmlReader حيث سيقوم هذا الغرض بقراءة الملف الحاوي على معلومات الطقس و الممثلة بالعناصر <city> ، <updated>، <forcast> ، <min>، <max>. سيكون ملف XML من الشكل:

```
<?xml version="1.0" ?>
<weatherinfo>
<auckland>
<updated>31/01/2002 09:00</updated>
<city>auckland</city>
<forecast>Fine. A mostly sunny day with light winds</forecast>
<min>24</min>
<max>25</max>
</auckland>
<chennai>
<updated>31/01/2002 09:00</updated>
<city>chennai</city>
<forecast>Fine.</forecast>
<min>35</min>
<max>38</max>
</chennai>
<hongkong>
<updated>31/01/2002 09:00</updated>
<city>hongkong</city>
<forecast>Early rain</forecast>
<min>23</min>
<max>25</max>
</hongkong>
<mumbai>
<updated>31/01/2002 09:00</updated>
<city>mumbai</city>
<forecast>Sunny day</forecast>
<min>29</min>
<max>35</max>
</mumbai>
<malaysia>
<updated>31/01/2002 09:00</updated>
```

```
<forecast>raining</forecast>
<min>24</min>
<max>25</max>
</malaysia>
<newdelhi>
<updated>31/01/2002 09:00</updated>
<city>newdelhi</city>
<forecast>Fine</forecast>
<min>30</min>
<max>35</max>
</newdelhi>
<newyork>
<updated>31/01/2002 09:00</updated>
<city>newyork</city>
<forecast>Very Cold</forecast>
<min>24</min>
<max>25</max>
</newyork>
<singapore>
<updated>31/01/2002 09:00</updated>
<city>singapore</city>
<forecast>rain day</forecast>
<min>31</min>
<max>35</max>
</singapore>
<tokyo>
<updated>31/01/2002 09:00</updated>
<city>tokyo</city>
<forecast>sunny day </forecast>
<min>22</min>
<max>25</max>
</tokyo>
<sydney>
<updated>31/01/2002 09:00</updated>
<city>sydney</city>
<forecast>Fine</forecast>
<min>21</min>
<max>25</max>
</sydney>
<washington>
<updated>31/01/2002 09:00</updated>
<city>washington</city>
<forecast>Fine.very cold</forecast>
<min>23</min>
<max>25</max>
</washington>
</weatherinfo>
//XML File End
```

إذا أردنا الآن كتابة النص البرمجي الخاص بقراءة وإظهار معلومات هذا الملف فسيكون من الشكل:

```
'Source Code Starts
<%@ Page Inherits=" System.Web.UI.MobileControls.MobilePage"Language="vb" %>
<%@ Register TagPrefix="mobile" Namespace="System.Web.UI.MobileControls"
Assembly="System.Web.Mobile" %>
<% @Import Namespace="System.Xml"%>

<script runat="server">
Public Sub List_ClickEventHandler(ByVal [source] As [Object], ByVal e As
ListCommandEventArgs)
```

```
Dim weatherReader As XmlTextReader = Nothing
weatherReader = New XmlTextReader(weatherFileName)
Dim selectedcity As [String] = e.ListItem.Value
While weatherReader.Read()
If weatherReader.NodeType = XmlNodeType.Element Then
If weatherReader.Name = e.ListItem.Value Then
WeatherLabel = e.ListItem.Text + ControlChars.Lf + "Weather"
If weatherReader.LocalName.Equals("updated") Then
WeatherLabel = WeatherLabel + ControlChars.Lf + weatherReader.ReadString()
End If
If weatherReader.LocalName.Equals("city") Then
WeatherLabel = WeatherLabel + weatherReader.ReadString()
End If
If weatherReader.LocalName.Equals("forecast") Then
WeatherLabel = WeatherLabel + weatherReader.ReadString()
End If
If weatherReader.LocalName.Equals("min") Then
WeatherLabel = WeatherLabel + "Min Temperature:" + weatherReader.ReadString()
End If
If weatherReader.LocalName.Equals("max") Then
WeatherLabel = WeatherLabel + "Max Temperature:" + weatherReader.ReadString()
End If
End If
End If
End While
ActiveForm = weather
End Sub 'List_ClickEventHandler
</script>
'List of cities
<mobile:Form runat="server">
<mobile:Label runat="server">Select a City</mobile:Label>
<mobile:List runat="server" id="Listcityvalue"
OnItemCommand="List_ClickEventHandler" >
<item Text="Auckland" Value="auckland" />
<item Text="Chennai" Value="chennai" />
<item Text="Hong Kong" Value="hongkong" />
<item Text="Mumbai" Value="mumbai" />
<item Text="Malaysia" Value="malaysia" />
<item Text="New Delhi" Value="newdelhi" />
<item Text="New York" Value="newyork" />
<item Text="Sydney" Value="sydney" />
<item Text="Singapore" Value="singapore" />
<item Text="Tokyo" Value="tokyo" />
<item Text="Washington" Value="washington" />
</mobile:List>
</mobile:Form>
<mobile:Form runat="server" id="SecondForm">
<mobile:Label runat="server" id="WelcomeMessage" />
</mobile:Form>
<mobile:Form id="weather" runat = "server">
<mobile:Label runat="server" id="WeatherLabel"/>
</mobile:Form>
'Source Code End
```

نلاحظ أننا قمنا بإدراج النص البرمجي كجزء من الصفحة دون فصله ضمن ملف منفصل.

المثال الثالث

سنقوم في هذا المثال بتخصيص إعدادات تطبيق بناء على المستخدم الذي سجل الدخول إلى هذا التطبيق، حيث يعد هذا الأسلوب فعالاً جداً للحصول على رضا الزبون وللتخفيف ما أمكن من ظهور معلومات غير مهمة لهذا المستخدم بشكل خاص.

يتألف مثالنا من تطبيق محمول لإظهار أسعار الأسهم حيث سيتمكن المستخدم من الوصول إلى أسعار الأسهم التي يختار رمزها.

يمكن للمستخدم حفظ الخيار الذي قام به وفي المرة التالية التي يسجل فيها دخوله سوف يتم إظهار معلومات هذه الأسهم بشكل له بشكل خاص.

سنقوم باستخدام قاعدة بيانات MS Access مع هذا المثال ونقوم باستخدام جدولين TblUser وTblStock سنستخدم في هذه المرة C#

TblUser	
UserID	Text
Pwd	Text

TblStock	
User ID	Text
StockSymbols	Text

يخزن الجدول TblStock المعلومات من الشكل مؤشرات الأسهم الخاصة بكل زبون وذلك بالشكل:

User ID	Stock Symbols
User1	MSFT
User2	CSCO,NT,XRX

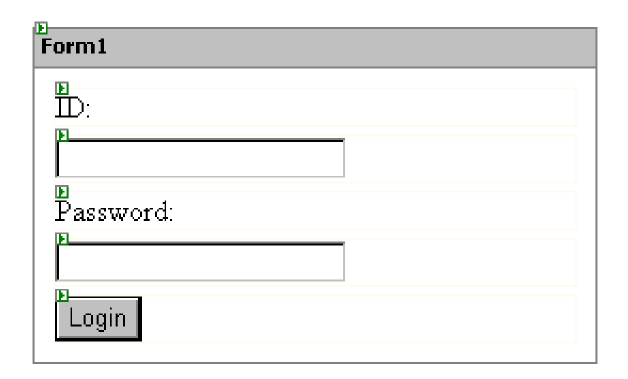
ملاحظة: بالطبع سيكون من المنطقي في نظام فعلي عدم تخزين معلومات كلمات السر بشكلها الصريح .

سيلزمنا أولاً إجراء بعض التغييرات إلى الملف web.config:

```
<authentication mode="Forms" >
<forms loginUrl="login.aspx" name=".ASPXCOOKIEAUTH" path="/">
</forms>
</authentication>
<authorization>
<deny users="?" />
</authorization>
```

التغييرات السابقة ستحول عملية التحقق من الهوية لتصبح مهمة النموذج في الصفحة login.aspx حيث يتم إدخال

اسم التسجيل وكلمة المرور وبعدها يقوم النموذج سيتم توجيه المستخدم من هذه الصفحة إلى الصفحة المطلوبة في حال تم قبول تسجيله. نبدأ أولاً بتصميم صفحة تسجيل الدخول.



و لما كنّا قد اخترنا استخدام قاعدة بيانات Access فسنقوم بجر عنصر التحكم الخاص بالاتصال بقواعد البيانات OleDbConnection وإنشاء اتصال يدل على قاعدة البيانات وذلك بضبط إعدادات ConnectionString.

بعدها سنقوم بجر عنصر التحكم الخاص Command أيضاً ونحدد خاصة SQL :

```
SELECT COUNT(UserID) AS Expr1 FROM tblUser WHERE (Pwd = ?) AND (UserID =?)
```

سيتم استخدام غرض Command للتحقق من المستخدم

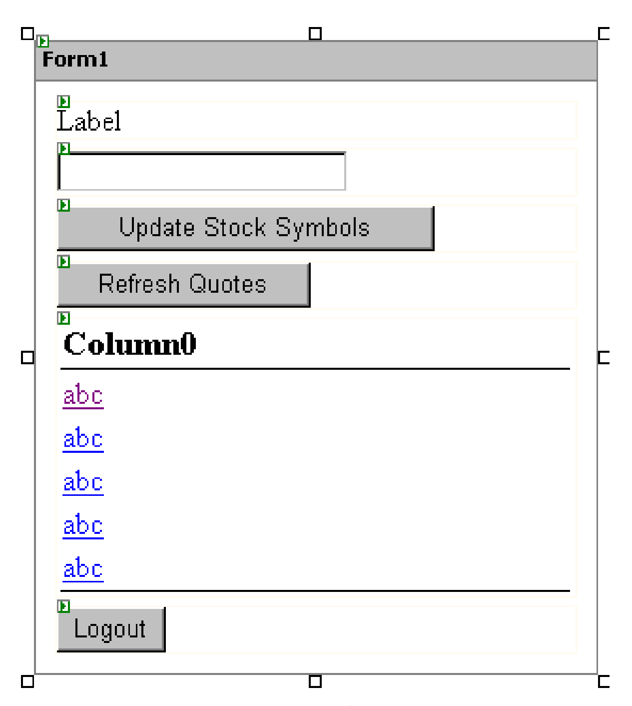
النص البرمجي التالي يوضح طريقة معالج الحدث الخاصة بالنقر على زر btnLogin.

```
Private Sub btnLogin_Click(ByVal sender As Object, ByVal e As System.EventArgs)
oleDbCommand1.Parameters.Add("Pwd", OleDbType.VarChar, 50)
oleDbCommand1.Parameters("Pwd").Value = txtPwd.Text
oleDbCommand1.Parameters.Add("UserID", OleDbType.VarChar, 50)
oleDbCommand1.Parameters("UserId").Value = txtUser.Text
oleDbConnection1.Open()
Dim nCount As Integer = CInt(oleDbCommand1.ExecuteScalar())
oleDbConnection1.Close()
If nCount = 1 Then
MobileFormsAuthentication.RedirectFromLoginPage(TextBox1.Text, True)
End If
End Sub 'btnLogin_Click
```

صفحة الوب المحمول:

ستتألف صفحتنا الأساسية هنا من نموذج وحيد يظهر فيه اسم المستخدم ورموز الأسهم وتظهر قيم أسعار الأسهم ضمن جدول.

يمكن للمستخدم تعديل تفضيلات الإعدادات المستخدمة.



العناصر في هذا النموذج وحسب الترتيب هي من النمط Label – TextBox-Button-Button-ObjectList

للتعرف على المستخدم سنقوم باستخدام الخاصة Context.User.Identity.Name

وبنفس الطريقة التي قمنا باستخدامها لتسجيل الدخول سنقوم بإضافة عنصر تحكم OleDbConnection وعنصر تحكم OleDbCommand ونحدد الخاصة CommandText إلى القيمة :

```
SELECT StockSymbols, UserId FROM tblStock WHERE (UserId = ?)
```

يأخذ النص البرمجي لتحميل الصفحة الشكل:

```
Private Sub Page_Load(ByVal sender As Object, ByVal e As System.EventArgs)
If Not IsPostBack Then
UserName = Context.User.Identity.Name
Label1.Text = UserName
TextBox1.Text = GetSymbolsForUser()
FillQuotes(TextBox1.Text)
End If
End Sub 'Page_Load
Code Snippet : Display the user's preferences and the values of the stock quotes.
Private Function GetSymbolsForUser() As String
oleDbCommand1.Parameters(0).Value = Context.User.Identity.Name
oleDbConnection1.Open()
Dim strSymbols As String = CStr(oleDbCommand1.ExecuteScalar())
oleDbConnection1.Close()
Return strSymbols
End Function 'GetSymbolsForUser
```

أما الطريقة التي ستقوم بإظهار أسعار الأسهم فهي التالية:

```
Private Sub FillQuotes(ByVal strSymbols As String)
Dim req As HttpWebRequest
Dim res As HttpWebResponse
Dim sr As StreamReader
Dim strResult As String
Dim temp() As String
Dim temp1() As String
```

```
Dim strcurindex As String
Dim fullpath As String
Dim ds As New DataSet
ds.Tables.Add("tblStk")
Dim SymbolColumn As New DataColumn
SymbolColumn.DataType = System.Type.GetType("System.String")
SymbolColumn.AllowDBNull = True
SymbolColumn.Caption = "Symbol"
SymbolColumn.ColumnName = "StkSymbol"
SymbolColumn.DefaultValue = "Stock"
' Add the column to the table.
ds.Tables("tblStk").Columns.Add(SymbolColumn)
'get stock quote for each row
Dim PriceColumn As New DataColumn
PriceColumn.DataType = System.Type.GetType("System.Decimal")
PriceColumn.AllowDBNull = True
PriceColumn.Caption = "Price"
PriceColumn.ColumnName = "StkPrice"
PriceColumn.DefaultValue = 0
' Add the column to the table.
ds.Tables("tblStk").Columns.Add(PriceColumn)
temp = strSymbols.Split(separator)
If temp.Length > 0 Then
Dim i As Integer
For i = 0 To temp.Length - 1
fullpath = "http://quote.yahoo.com/d/quotes.csv?s=" + temp(i) +
"&f=sl1d1t1c1ohgvj1pp2owern&e=.csv"
Try
req = CType(WebRequest.Create(fullpath), HttpWebRequest)
res = CType(req.GetResponse(), HttpWebResponse)
sr = New StreamReader(res.GetResponseStream(), Encoding.ASCII)
strResult = sr.ReadLine()
sr.Close()
temp1 = strResult.Split(separator)
If temp1.Length > 1 Then
'only the relevant portion.
strcurindex = temp1(1)
Dim myRow As DataRow = ds.Tables("tblStk").NewRow()
myRow(0) = temp(i)
myRow(1) = Convert.ToDecimal(strcurindex)
ds.Tables("tblStk").Rows.Add(myRow)
End If
Catch
End Try
Next i
ObjectList1.DataSource = ds.Tables("tblStk").DefaultView
ObjectList1.DataBind()
ObjectList1.TableFields = "StkSymbol;StkPrice"
End If
End Sub 'FillQuotes
```

أما الطريقة الخاصة بمعالجة حدث نقر زر Refresh Quote

```
Private Sub Command3_Click(ByVal sender As Object, ByVal e As System.EventArgs)
FillQuotes(GetSymbolsForUser())
End Sub 'Command3_Click
```

ولتحديث الرموز الخاصة بالمستخدم نلجأ إلى استخدام نفس الطريقة المتبعة سابقاً للاتصال بقاعدة البيانات باستخدام غرض Connection وغرض Command مع تحديد قيمة CommandText إلى القيمة

```
UPDATE tblStock SET StockSymbols = ? WHERE (UserId = ?)
```

نقوم هنا كما نلاحظ بتمرير المعاملات لاستعلام SQL وذلك لتحديث الرموز الخاصة بالأسهم المفضلة لدى المستخدم.

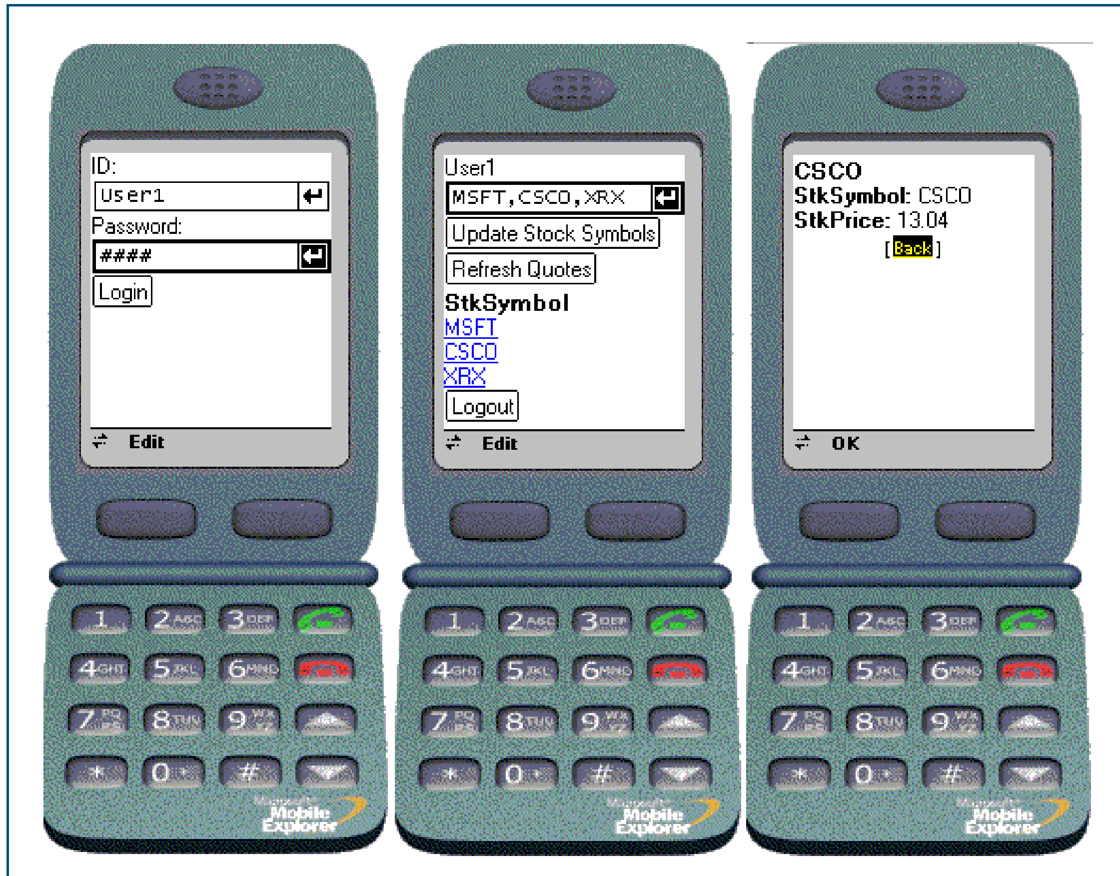
```
Private Sub Command2_Click(ByVal sender As Object, ByVal e As System.EventArgs)
oleDbCommand2.Parameters(0).Value = TextBox1.Text
oleDbCommand2.Parameters(1).Value = Context.User.Identity.Name
oleDbConnection1.Open()
oleDbCommand2.ExecuteNonQuery()
oleDbConnection1.Close()
FillQuotes(GetSymbolsForUser())
End Sub 'Command2_Click
```

أما النص البرمجي الخاص بمعالج حدث ضغط زر تسجيل الخروج فهو كما يلي:

```
Private Sub Command1_Click(ByVal sender As Object, ByVal e As System.EventArgs)
MobileFormsAuthentication.SignOut()
RedirectToMobilePage("login.aspx")
End Sub 'Command1_Click
```

أما مصدر بيانات أسعار الأسهم فسيتم استجلابها من موقع Yahoo بحسب ما يرد في النص الكامل للبرنامج و ذلك من المسار التالي.

```
fullpath = "http://quote.yahoo.com/d/quotes.csv?s=" + temp(i) +
"&f=sl1d1t1c1ohgvj1pp2owern&e=.csv"
```



فيما يلي النص الكامل للبرنامج :

الملف Default.aspx

```
<%@ Page Inherits="System.Web.UI.MobileControls.MobilePage" Language="VB"
Debug="true" %>
<%@ Register TagPrefix="mobile" Namespace="System.Web.UI.MobileControls"
Assembly="System.Web.Mobile" %>
<%@ Import Namespace="System" %>
<%@ Import Namespace="System.Data"%>
<%@ Import Namespace="System.Data.OleDb" %>
<%@ Import Namespace="System.Net" %>
<%@ Import Namespace="System.Text" %>
<%@ Import Namespace="System.IO" %>
<%@ Import Namespace="System.Web.Mobile" %>
<script runat="server" language="VB">
Public str As [String]
Public strConn As String = "Provider=Microsoft.Jet.OLEDB.4.0;Data
Source=C:\inetpub\wwwroot\Prsnlmob\db1.mdb"
Private separator As Char() = ","c
Protected oleDbConnection1 As OleDbConnection
Protected oleDbCommand1 As OleDbCommand
Protected oleDbCommand2 As OleDbCommand
Private strUserName As String
Private Sub Page_Load(ByVal sender As Object, ByVal e As System.EventArgs)
oleDbConnection1 = New OleDbConnection(strConn)
oleDbCommand1 = New OleDbCommand
oleDbCommand2 = New OleDbCommand
If Not IsPostBack Then
strUserName = Context.User.Identity.Name
Label1.Text = strUserName
TextBox1.Text = GetSymbolsForUser()
```

```
End If
End Sub 'Page_Load
Private Sub Command1_Click(ByVal sender As Object, ByVal e As System.EventArgs)
'Signout
MobileFormsAuthentication.SignOut()
RedirectToMobilePage("login.aspx")
End Sub 'Command1_Click
Private Sub FillQuotes(ByVal strSymbols As String)
'this function will fetch stock quotes for each stock symbol specified by the user
and populate the data in a datatable. The data is finally bound to an ObjectList.
Dim req As HttpWebRequest
Dim res As HttpWebResponse
Dim sr As StreamReader
Dim strResult As String
Dim temp() As String
Dim temp1() As String
Dim strcurindex As String
Dim fullpath As String
Dim ds As New DataSet
ds.Tables.Add("tblStk")
Dim SymbolColumn As New DataColumn
SymbolColumn.DataType = System.Type.GetType("System.String")
SymbolColumn.AllowDBNull = True
SymbolColumn.Caption = "Symbol"
SymbolColumn.ColumnName = "StkSymbol"
SymbolColumn.DefaultValue = "MSFT"
' Add the column to the table.
ds.Tables("tblStk").Columns.Add(SymbolColumn)
'get stock quote for each row
Dim PriceColumn As New DataColumn
PriceColumn.DataType = System.Type.GetType("System.Decimal")
PriceColumn.AllowDBNull = True
PriceColumn.Caption = "Price"
PriceColumn.ColumnName = "StkPrice"
PriceColumn.DefaultValue = 0
' Add the column to the table.
ds.Tables("tblStk").Columns.Add(PriceColumn)
temp = strSymbols.Split(separator)
If temp.Length > 0 Then
Dim i As Integer
For i = 0 To temp.Length - 1
fullpath = "http://quote.yahoo.com/d/quotes.csv?s=" + temp(i) +
"&f=sl1d1t1c1ohgvj1pp2owern&e=.csv"
'
Try
req = CType(WebRequest.Create(fullpath), HttpWebRequest)
res = CType(req.GetResponse(), HttpWebResponse)
sr = New StreamReader(res.GetResponseStream(), Encoding.ASCII)
strResult = sr.ReadLine()
sr.Close()
temp1 = strResult.Split(separator)
If temp1.Length > 1 Then
'only the relevant portion .
strcurindex = temp1(1)
Dim myRow As DataRow = ds.Tables("tblStk").NewRow()
myRow(0) = temp(i)
myRow(1) = Convert.ToDecimal(strcurindex)
ds.Tables("tblStk").Rows.Add(myRow)
End If
Catch
End Try
```

```
ObjectList1.DataSource = ds.Tables("tblStk").DefaultView
ObjectList1.DataBind()
ObjectList1.TableFields = "StkSymbol;StkPrice"
End If
End Sub 'FillQuotes
Private Sub Command2_Click(ByVal sender As Object, ByVal e As System.EventArgs)
'the following code will update the Stock Symbol preferences as specified by the
user.
oleDbCommand2.Connection = oleDbConnection1
oleDbCommand2.CommandText = "UPDATE tblStock SET StockSymbols = ? WHERE
UserId = ?)"
oleDbCommand2.Parameters.Add(New System.Data.OleDb.OleDbParameter("StockSymbols",
System.Data.OleDb.OleDbType.VarWChar, 255, "StockSymbols"))
oleDbCommand2.Parameters.Add(New System.Data.OleDb.OleDbParameter("Original_UserId",
System.Data.OleDb.OleDbType.VarWChar, 50, System.Data.ParameterDirection.Input,
False, CType(0, System.Byte), CType(0, System.Byte), "UserId",
System.Data.DataRowVersion.Original, Nothing))
oleDbCommand2.Parameters(0).Value = TextBox1.Text
oleDbCommand2.Parameters(1).Value = Context.User.Identity.Name
oleDbConnection1.Open()
oleDbCommand2.ExecuteNonQuery()
oleDbConnection1.Close()
FillQuotes(GetSymbolsForUser())
End Sub 'Command2_Click
Private Sub Command3_Click(ByVal sender As Object, ByVal e As System.EventArgs)
'Refresh the stock quotes
FillQuotes(GetSymbolsForUser())
End Sub 'Command3_Click
Private Function GetSymbolsForUser() As String
'Fetch the preferences specified by the user from the database
oleDbCommand1.Connection = oleDbConnection1
oleDbCommand1.CommandText = "SELECT StockSymbols, UserId FROM tblStock WHERE (UserId
?)"
Me.oleDbCommand1.Parameters.Add(New System.Data.OleDb.OleDbParameter("UserId",
System.Data.OleDb.OleDbType.VarWChar, 50, "UserId"))
oleDbCommand1.Parameters(0).Value = Context.User.Identity.Name
oleDbConnection1.Open()
Dim strSymbols As String = CStr(oleDbCommand1.ExecuteScalar())
oleDbConnection1.Close()
Return strSymbols
End Function 'GetSymbolsForUser
</script>
<mobile:Form id = "Form1" runat="server">
<mobile:Label id="Label1" runat="server">Label</mobile:Label>
<mobile:TextBox id="TextBox1" runat="server"></mobile:TextBox>
<mobile:Command id="Command2" runat="server"
onClick="Command2_Click">Update Stock Symbols</mobile:Command>
<mobile:Command id="Command3" runat="server" onClick="Command3_Click">Refresh
Quotes</mobile:Command>
<mobile:ObjectList id="ObjectList1" runat="server" LabelStyle-StyleReference="title"
CommandStyle-StyleReference="subcommand"></mobile:ObjectList>
<mobile:Command id="Command1" runat="server"
OnClick="Command1_Click">Logout</mobile:Command>
</mobile:Form>
```

والملف Login.aspx

```
<%@ Page Inherits="System.Web.UI.MobileControls.MobilePage" Language="VB"
Debug="true" %>
<%@ Assembly Name="System.Web" %>
```

```
Assembly="System.Web.Mobile" %>
<%@ Import Namespace="System" %>
<%@ Import Namespace="System.Data"%>
<%@ Import Namespace="System.Data.OleDb" %>
<%@ Import Namespace="System.web" %>
<%@ Import Namespace="System.web.Security" %>
<%@ Import Namespace="System.Web.Mobile" %>
<script runat="server" language="VB">
Public str As [String]
Public strConn As String = "Provider=Microsoft.Jet.OLEDB.4.0;Data Source=C:\
inetpub\wwwroot\Prsnl\db1.mdb"
Protected oleDbCommand1 As OleDbCommand
Private Sub Page_Load(ByVal sender As Object, ByVal e As System.EventArgs)
oleDbConnection1 = New OleDbConnection(strConn)
oleDbCommand1 = New OleDbCommand
End Sub 'Page_Load
Private Sub Command1_Click(ByVal sender As Object, ByVal e As System.EventArgs)
'login using the credentials specified by the user
oleDbCommand1.Connection = oleDbConnection1
oleDbCommand1.CommandText = "SELECT COUNT(UserID) AS Expr1 FROM tblUser WHERE (Pwd =
?) AND (UserID = ?)"
oleDbCommand1.Parameters.Add("UserId", OleDbType.VarChar, 50)
oleDbCommand1.Parameters(0).Value = txtPwd.Text
oleDbCommand1.Parameters.Add("UserId", OleDbType.VarChar, 50)
oleDbCommand1.Parameters(1).Value = txtUser.Text
oleDbConnection1.Open()
Dim nCount As Integer = CInt(oleDbCommand1.ExecuteScalar())
oleDbConnection1.Close()
If nCount >= 1 Then
MobileFormsAuthentication.RedirectFromLoginPage(txtUser.Text, True)
End If
End Sub 'Command1_Click
</script>
<mobile:Form id = "Form1" runat="server">
<mobile:Label id="Label1" runat="server">ID:</mobile:Label>
<mobile:TextBox id="txtUser" runat="server"></mobile:TextBox>
<mobile:Label id="Label2" runat="server">Password:</mobile:Label>
<mobile:TextBox id="txtPwd" runat="server" Password="True"></mobile:TextBox>
<mobile:Command id="cmdLogin" runat="server"
onClick="Command1_Click">Login</mobile:Command>
</mobile:Form>
```

وسيأخذ الملف Web.config بعد التعديل الشكل :

```
<?xml version="1.0" encoding="utf-8" ?>
<configuration>
<system.web>
<compilation defaultLanguage="VB" debug="true"/>
<customErrors mode="Off" />
<authentication mode="Forms" >
<forms loginUrl="login.aspx" name=".ASPXCOOKIEAUTH" path="/">
</forms>
</authentication>
<authorization>
<deny users="?" />
</authorization>
<trace enabled="false" requestLimit="10" pageOutput="false" traceMode="SortByTime"
localOnly="true"/>
```

```
sqlConnectionString="data source=127.0.0.1;user id=sa;password=" cookieless="true"
timeout="20" />
<globalization requestEncoding="utf-8" responseEncoding="utf-8" />
| <httpRuntime useFullyQualifiedRedirectUrl="true" />
<mobileControls cookielessDataDictionaryType="System.Web.Mobile.CookielessData" />
<deviceFilters>
<filter name="isHTML32" compare="PreferredRenderingType" argument="html32" />
<filter name="isWML11" compare="PreferredRenderingType" argument="wml11" />
<filter name="isCHTML10" compare="PreferredRenderingType" argument="chtml10" />
<filter name="isGoAmerica" compare="Browser" argument="Go.Web" />
<filter name="isMME" compare="Browser" argument="Microsoft Mobile Explorer" />
<filter name="isMyPalm" compare="Browser" argument="MyPalm" />
<filter name="isPocketIE" compare="Browser" argument="Pocket IE" />
<filter name="isUP3x" compare="Type" argument="Phone.com 3.x Browser" />
<filter name="isUP4x" compare="Type" argument="Phone.com 4.x Browser" />
<filter name="isEricssonR380" compare="Type" argument="Ericsson R380" />
<filter name="isNokia7110" compare="Type" argument="Nokia 7110" />
<filter name="prefersGIF" compare="PreferredImageMIME" argument="image/gif" />
<filter name="prefersWBMP" compare="PreferredImageMIME" argument="image/vnd.wap.wbmp"
/>
<filter name="supportsColor" compare="IsColor" argument="true" />
<filter name="supportsCookies" compare="Cookies" argument="true" />
<filter name="supportsJavaScript" compare="Javascript" argument="true" />
<filter name="supportsVoiceCalls" compare="CanInitiateVoiceCall" argument="true" />
</deviceFilters>
</system.web>
</configuration>
```

قراءات اضافية:

1	http://www.w3schools.com/dotnetmobile/default.asp
2	http://www.aspnextgen.com/MobileQuickStart/(qz3zc5ygksjxka45e4wwv2uf)/Default.aspx
3	http://www.asp.net/default.aspx?tabIndex=3&tabId=44
4	http://www.wirelessdevnet.com/channels/wap/
5	http://www.microsoft.com/downloads/details.aspx?FamilyID=8fb566e0-3e92-40e8-b5d4-091d05ab8829&DisplayLang=en
6	http://www.ondotnet.com/pub/a/dotnet/2004/02/23/mobilewebserviceapps.html